Verwenden von ADOBE® FLASH® BUILDER™ 4.6

Rechtliche Hinweise

Rechtliche Hinweise finden Sie unter http://help.adobe.com/de_DE/legalnotices/index.html.

Letzte Aktualisierung 9.12.2011

Inhalt

Kapitel 1: Flash Builder
Anwendungen, die mit Flash Builder erstellt werden können ... 1
Funktionen zur Erleichterung der Anwendungsentwicklung ... 2
Funktionen zum Ausführen und Debuggen von Projekten ... 3
Flash Builder-Versionen ... 3
Flash Builder-Konfigurationen ... 4
Adobe Community Help Client (CHC) ... 4

Kapitel 2: Erste Schritte mit Flash Builder
Flash Builder-Workbench ... 6
Flash Builder-Perspektiven und -Ansichten ... 7
Editoren in Flash Builder ... 21
Projekte in Flash Builder ... 35
Projektressourcen ... 42

Kapitel 3: Tools zur Codeentwicklung in Flash Builder
Inhaltshilfe ... 49
Schnellhilfe ... 53
Methoden überschreiben oder implementieren ... 56
Codevorlagen ... 57
Metadaten-Codevervollständigung ... 61
Dateivorlagen anpassen ... 63
Aus der Verwendung generieren ... 72
Get-/Set-Accessor-Funktionen generieren ... 74
Auf Syntaxfehler prüfen ... 75
Unbekannte Verweise hervorheben ... 77
Verweise suchen und Code refaktorieren ... 77
Formatieren, navigieren und Code organisieren ... 78

Kapitel 4: Projekte in Flash Builder
Projekte in Flash Builder erstellen ... 90
Projekte exportieren und importieren ... 97
Projekte erstellen ... 103
Anwendungen ausführen und debuggen ... 123
Releaseversion einer Anwendung exportieren ... 129
AIR-Anwendungen verpacken ... 134

Kapitel 5: Debugging-Tools in Flash Builder
Flash-Debugperspektive ... 138
Anleitung zum Debuggen einer Anwendung ... 141

Kapitel 6: Profiling-Tools in Flash Builder
Flash-Profilingperspektive ... 149
Profiling einer Anwendung ... 163

Funktionsweise des Profilers ... 164
Der Profiler ... 166
Garbage Collection ... 172
Problembereiche ermitteln ... 173
Profiler-Filter ... 176

Kapitel 7: Unit-Test-Tools in Flash Builder
FlexUnit-Testumgebung ... 178
FlexUnit-Tests erstellen ... 178
FlexUnit-Tests ausführen ... 180
FlexUnit-Tests konfigurieren ... 182
Ergebnisse eines FlexUnit-Testlaufs anzeigen ... 183
FlexUnit-Unterstützung für Mobilprojekte ... 184

Kapitel 8: Entwickeln von Anwendungen mit Flex
Einfacher Arbeitsablauf der Anwendungsentwicklung mit Flex ... 185
Benutzeroberflächen erstellen ... 186
Ereignisprozeduren generieren ... 218
Auf Datendienste zugreifen ... 221
Anwendungen überwachen, die auf Datendienste zugreifen ... 230
Flex-Bibliotheksprojekte ... 234
Benutzerdefinierte MXML-Komponenten erstellen ... 239
Module erstellen ... 241
Flex in HTML-Anwendungen integrieren ... 246

Kapitel 9: Verwenden von Flash Builder mit Flash Professional
Flash Professional-Projekt erstellen ... 251
Was man in einem Flash Professional-Projekt machen kann ... 251
Projekteigenschaften für Flash Professional-Projekte festlegen ... 252
Flash-Komponenten erstellen und bearbeiten ... 252
Flash Player-Sicherheit verwalten ... 254

Kapitel 10: Verwenden von Flash Builder mit Flash Catalyst
Arbeitsabläufe zwischen Flash Catalyst und Flash Builder ... 257
Integration von Flash Catalyst und Flash Builder ... 261

Kapitel 11: Anpassen von Flash Builder
Adobe-Voreinstellungen ... 266
Workbench anpassen ... 266
Flash Builder-Voreinstellungen ... 268
Flash Builder erweitern ... 278

Kapitel 1: Flash Builder

Adobe® Flash® Builder™ ist eine integrierte Entwicklungsumgebung (IDE, Integrated Development Environment) zum Erstellen von plattformübergreifenden Rich-Internet-Applications (RIAs) für Desktop-Computer und eine Reihe von Mobilgeräten. Flash Builder enthält auch Werkzeuge für Test, Debugging und Profiling, die zur Produktivitäts- und Effizienzsteigerung eingesetzt werden können.

Flash Builder setzt auf Eclipse (einer Open-Source-IDE) auf und stellt alle Werkzeuge zur Verfügung, die Sie zum Entwickeln von Anwendungen, die das Flex-Framework und ActionScript 3.0 nutzen, benötigen. Flash Builder läuft unter Microsoft Windows und Apple Mac OS X und ist in mehreren Versionen verfügbar. Flash Builder kann entweder als Plug-In-Satz in einer bereits installierten Eclipse-Workbench oder als eigenständige Installation, die die Eclipse-Workbench enthält, installiert werden.

Anwendungen, die mit Flash Builder erstellt werden können

Sie können Anwendungen erstellen, die das Adobe Flex®-Framework, MXML, Adobe Flash Player, Adobe AIR, ActionScript 3.0, und ADEP (Adobe Digital Enterprise Platform) Data Services und die Adobe® Flex®-Diagrammkomponenten nutzen:

- **Flex-Projekte** – Erstellen Sie Flex-Projekte, die mit beliebigen Backend-Servertechnologien funktionieren, z. B. Adobe ColdFusion®, ADEP Data Services und PHP.

  Siehe „Flex-Projekte“ auf Seite 90 und „Entwickeln von Anwendungen mit Flex“ auf Seite 185.

- **ActionScript-Projekte** – Erstellen Sie ActionScript-Projekte mit Flash-API (statt mit Flex-Framework).

  Siehe „ActionScript-Projekte“ auf Seite 93.

- **Mobilprojekte** – Erstellen Sie Flex-Mobilanwendungen für die Google Android-Plattform und ActionScript-Mobilanwendungen für die Apple iOS-Plattform.

  Siehe „Flex-Mobilprojekte“ auf Seite 92 und „ActionScript-Mobilprojekte erstellen“ auf Seite 93.

  Weitere Informationen zur Entwicklung von Mobilanwendungen mit Flex und Flash Builder finden Sie unter Entwickeln von Mobilanwendungen mit Flex und Flash Builder.

- **Bibliotheksprojekte** – Erstellen Sie individuelle Codebibliotheken, die Sie als Komponentenblibliotheksdateien (SWC-Dateien) zur gemeinsamen Nutzung in Ihren Anwendungen bereitstellen oder an andere Entwickler verteilen können.

  Siehe „Flex-Bibliotheksprojekte“ auf Seite 234.

- **Flash Professional-Projekte** – Erstellen Sie Flash Professional-Projekte, um mit Flash Professional CS5 erstellte Flash FLA- oder XFL-Dateien zu bearbeiten oder zu debuggen.

  Siehe „Verwenden von Flash Builder mit Flash Professional“ auf Seite 251.

- **Mit Flash Catalyst kompatible Projekte** – Erstellen Sie mit Flash Catalyst kompatible Projekte, an denen Designer und Entwickler zusammenarbeiten können. Mithilfe von FXP- oder FXPL-Dateien können Sie ein mit Flash kompatibles Projekt sowohl in Flash Builder als auch in Flash Catalyst nutzen.

  Siehe „Mit Flash Catalyst kompatible Projekte“ auf Seite 94.

Informationen zu den Workflows zwischen Flash Builder und Flash Catalyst finden Sie im Artikel Die neuen Workflows in Flash Catalyst CS und Flash Builder des Adobe-Produkt-Managers Jacob Surber.

- **Benutzerdefinierte MXML-Komponenten** – Erstellen Sie benutzerdefinierte Komponenten, die dann in der Komponentenansicht von Flash Builder zur Verfügung stehen.

  Siehe „Benutzerdefinierte MXML-Komponenten erstellen“ auf Seite 239.

  Sie können auch benutzerdefinierte ActionScript-Komponenten erstellen. Siehe „ActionScript-Klasse erstellen“ auf Seite 43.

Funktionen zur Erleichterung der Anwendungsentwicklung

FlashBuilder bietet Ihnen Werkzeuge zur Entwicklung von Anwendungen, die auf dem Flex-Framework und ActionScript 3.0 basieren. Sie können:

- **Auf Datendienste zugreifen** – Mit den Werkzeugen und Assistenten von Flash Builder können Sie Anwendungen erstellen, die auf Remote-Datendienste zugreifen.

  Siehe Überblick über den Zugriff auf Datendienste.

- **Die FlashBuilder-Workbench anpassen** – Passen Sie die Workbench an Ihre individuellen Enticklungsbedürfnisse an. Beispielsweise können Sie die Oberfläche so gestalten, dass die am häufigsten verwendeten Werkzeuge im jeweiligen Layout enthalten sind.

  Siehe „Workbench anpassen“ auf Seite 266.

- **Code bearbeiten** – Programmieren Sie Ihren Anwendungsquellcode mit den Codeentwicklungstools in Flash Builder. Zu den Codeentwicklungstools gehören u. a.: Coderefakturierung, Codehinweise, gestraffte Codenavigation, und automatische Syntaxfehlerprüfung.

  Siehe „Tools zur Codeentwicklung in Flash Builder“ auf Seite 49.

- **Benutzeroberflächen erstellen** – Entwickeln Sie mit dem MXML-Editor Anwendungen im Designmodus. In diesem Modus können Sie mit Layoutoptionen arbeiten, Komponenten auf die Designfläche ziehen und sie dann nach Bedarf umpositionieren und skalieren, Ansichtsstatus einfacher anzeigen usw.

  Siehe „Benutzeroberflächen erstellen“ auf Seite 186.

- **Quellcode veröffentlichen** – Veröffentlichen Sie Ihren Anwendungsquellcode, damit auch Benutzer und andere Entwickler ihn einsehen können.

  Siehe „Quellcode veröffentlichen“ auf Seite 121.

- **Projekte, Ordner, Dateien und andere Ressourcen verwalten** – Erstellen, ändern und löschen Sie Projekte und Ressourcen, erstellen Sie Verknüpfungen zu externen Ressourcen usw.

  Siehe „Projekte in Flash Builder“ auf Seite 35 und „Projektressourcen“ auf Seite 42.

Funktionen zum Ausführen und Debuggen von Projekten

Flash Builder enthält Werkzeuge zum Erstellen, Testen, Debuggen und Profiling, mit denen Sie die Arbeitsproduktivität erhöhen:

- **Projekte erstellen** – Flash Builder kompiliert und erstellt Ihre Anwendungen automatisch für das Debugging- oder das Produktions-Stadium. Mit Apache Ant können Sie zudem individuelle Erstellungsskripte erzeugen.

  Siehe „Projekte erstellen“ auf Seite 103.

- **Anwendungen ausführen und Startkonfigurationen verwalten** – Führen Sie Ihre Anwendungen im Webbrowser, mit AIR oder im eigenständigen Flash Player aus. Erstellen Sie benutzerdefinierte Startkonfigurationen, um zu steuern, wie Anwendungen ausgeführt werden.

  Siehe „Anwendungen ausführen und debuggen“ auf Seite 123 und „Startkonfigurationen verwalten“ auf Seite 123.

- **Anwendungen debuggen** – Debuggen Sie Ihre Anwendungen mit den integrierten Debugwerkzeugen.

  Siehe „Debugging-Tools in Flash Builder“ auf Seite 138.

- **Mobilanwendungen ausführen und debuggen** – Sie können Ihre Anwendungen auf dem Desktop oder auf einem Gerät ausführen und debuggen.

  Siehe Ausführen und Debuggen von Mobilanwendungen.

- **Profilanwendungen** – Identifizieren mithilfe der Profiling-Tools in Flash Builder Leistungsengpässe und Speicherlecks.

  Siehe „Profiling-Tools in Flash Builder“ auf Seite 149.

- **Anwendungen überwachen, die auf Datendienste zugreifen** – Mit der Netzwerküberwachung erzeugen Sie ein ausführliches Prüfprotokoll zum gesamten Datenverkehr zwischen Ihrer Anwendung und dem Backend.

  Siehe „Anwendungen überwachen, die auf Datendienste zugreifen“ auf Seite 230.

Flash Builder-Versionen

Flash Builder ist in zwei Versionen erhältlich: Standard und Premium.

Flash Builder Standard Flash Builder Standard enthält eine IDE mit allen Funktionen, die zum Erstellen von Anwendungen mit dem Flex-Framework und der Flash-API benötigt werden. Flash Builder Standard bietet außerdem MXML-, ActionScript- und CSS-Editoren sowie Debugwerkzeuge. Flash Builder Standard enthält eine Bibliothek mit interaktiven Diagrammen, über die Sie Rich-Data-Dashboards, interaktive Datenanalysen und Datenvisualisierungskomponenten erstellen können.

Flash Builder Premium Flash Builder Premium bietet zusätzlich zu den Funktionen der Standardversion Speicher- und Leistungs-Profiling sowie automatisierte Testwerkzeuge. Mit der Netzwerküberwachung können Sie Daten anzeigen lassen, die zwischen einer Clientanwendung und einem Datendienst ausgetauscht werden. In der FlexUnit-Testumgebung können Sie wiederholbare Tests generieren und bearbeiten. Der wiederholbare Test kann von Skripten aus, direkt im Flash Builder oder außerhalb der Flash Builder-Umgebung ausgeführt werden. Mit den Befehlszeilenerstellungsfunktionen können Sie die individuellen Erstellungseinstellungen eines Entwicklers mit einem nächtlichen Erstellungsprozess synchronisieren.

Flash Builder für PHP Standard Flash Builder für PHP Standard enthält eine PHP-IDE mit allen Funktionen, einschließlich Flash Builder Standard integriert mit Zend Studio 8. Sie können Mobil-, Web- und Desktopanwendungen mithilfe von PHP, Flex und ActionScript erstellen.

Flash Builder für PHP Premium Flash® Builder™ für PHP Premium bietet zusätzlich zu den Funktionen der Standardversion professionelle Testwerkzeuge, einschließlich Profiler, Netzwerküberwachung, eines automatisierten Test-Frameworks, Integration von FlexUnit-Tests und Unterstützung von Befehlszeilenerstellungsprozessen.

Informationen zu Flash Builder für PHP finden Sie unter Introducing Flash Builder 4.5 for PHP.

Flash Builder-Konfigurationen

Flash Builder bietet ein einheitliches Installationsprogramm mit den folgenden beiden Konfigurationsoptionen:

Eigenständige Konfiguration Installiert Flash Builder als eigenständige integrierte Entwicklungsumgebung (Integrated Development Environment, IDE). Die eigenständige Konfiguration ist speziell zum Entwickeln von Anwendungen mit dem Flex-Framework und ActionScript 3.0 vorgesehen. Diese Konfiguration ist ideal für Einsteiger sowie Benutzer, die Anwendungen ausschließlich mit dem Flex-Framework und ActionScript 3.0 entwickeln möchten.

Plug-In-Konfiguration Konfiguriert Flash Builder für die Ausführung als Plug-In für eine Eclipse™-Installation. Für die Ausführung der Plug-In-Konfiguration muss Eclipse 3.6 32-Bit (Windows) oder Eclipse 3.6 Cocoa 32-Bit (Mac) oder eine höhere Version von Eclipse auf Ihrem Computer installiert sein.

Beide Konfigurationen (eigenständige und Flash Builder-Plug-In) bieten die gleichen Funktionen. Wenn Sie sich nicht sicher sind, welche Konfiguration Sie verwenden sollen, beachten Sie bitte die folgenden Richtlinien:

- Wenn Sie bereits Eclipse 3.6 oder höher installiert haben, verwenden Sie die Plug-In-Konfiguration, um Ihrer vorhandenen Eclipse-Version Flash Builder-Funktionen hinzuzufügen.
- Wenn Sie Eclipse nicht installiert haben und primär Flex- und ActionScript-Anwendungen entwickeln möchten, verwenden Sie die eigenständige Flash Builder-Konfiguration. Diese Konfiguration ermöglicht es Ihnen auch, weitere Eclipse-Plug-Ins zu installieren, damit Sie zu einem späteren Zeitpunkt ggf. den Umfang Ihrer Entwicklungsmöglichkeiten erweitern können.

Ausführliche Informationen zum Installieren von Flash Builder finden Sie in den Versionshinweisen zu Flash Builder 4.6.

Adobe Community Help Client (CHC)

Flash Builder verwendet den Adobe Community Help Client (CHC) als Suchmaschine. CHC ist eine AIR-basierte Anwendung, die das Eclipse-Hilfemodul für Flash Builder ersetzt und die Plattform für die nächste Generation der Adobe-Hilfe bildet. CHC weist u. a. die folgenden Merkmale auf:

- **Immer online**

  Wenn Sie mit dem Internet verbunden sind, greift CHC über das Web auf die Hilfe zu. CHC stellt sicher, dass Sie auf das aktuellste Material zugreifen. Fehlt die Internetverbindung, funktioniert die Hilfe auch im lokalen Modus.

- **Suche als zentrale Funktion**

  Sie können die Community Help-Suche, adobe.com-Suche oder lokale Suche verwenden. Die Community Help-Suche stellt Ressourcen unter Einbeziehung von Inhalten unabhängiger Websites zusammen. Die adobe.com-Suche bietet Verfeinerungsmöglichkeiten, um die Suche einzuschränken.

- **Kontextbezogene Navigation**

  Stellt dynamisch generierte verwandte Links zu wichtigen Seiten zur Verfügung.

- **Bewertung und Kommentare**

  Kommentieren und bewerten Sie den Inhalt der Hilfe und der Onlineressourcen von CHC aus und steuern Sie eigene Beiträge bei. Für die Abgabe von Feedbacks und Bewertungen benötigen Sie eine gültige Adobe-ID.

Kapitel 2: Erste Schritte mit Flash Builder

Flash Builder-Workbench

Bei der Flash Builder-Workbench handelt es sich um eine ausgewachsene Entwicklungsumgebung zur Entwicklung von Anwendungen, die auf dem Adobe Flex®-Framework und auf ActionScript 3.0 basieren. Sie können Anwendungen für Adobe Flash Player, Desktop-Anwendungen für Adobe AIR® und Mobilanwendungen für diverse Mobilgeräte entwickeln.

Flash Builder wird in der zum Programm gehörenden Dokumentation eingehend beschrieben. Wenn Sie andere Eclipse-Plug-Ins (wie CVS oder Java) mit Flash Builder einsetzen oder wenn Sie die Flash Builder-Plug-Ins erweitern möchten, siehe Adobe Flash Builder Extensibility Reference.

Workbench Der Ausdruck *Workbench* bezieht sich auf die Flash Builder-Entwicklungsumgebung. Sie enthält alle Tools zur Anwendungsentwicklung. Die Workbench besteht aus drei Hauptkomponenten: *Perspektiven*, *Editoren* und *Ansichten*. Bei der Entwicklung einer Anwendung verwenden Sie alle drei Komponenten zu verschiedenen Zeitpunkten und in verschiedenen Kombinationen.

Hinweis: *Weitere Informationen zu einigen Funktionen der Eclipse-Workbench finden Sie im Benutzerhandbuch zur Eclipse-Workbench unter http://help.eclipse.org/help31/index.jsp.*

Perspektive Eine Perspektive ist eine Gruppe von Ansichten und Editoren in der Workbench. Flash Builder besteht aus zwei Perspektiven: Flash-Entwicklungsperspektive zur Anwendungsentwicklung und Flash-Debugperspektive zum Debuggen von Anwendungen. Flash Builder Premium enthält zusätzlich noch die Flash-Profilingperspektive.

Wenn Sie die Flash Builder-Plug-In-Konfiguration verwenden (siehe „Flash Builder-Konfigurationen“ auf Seite 4), enthält die Workbench u. U. weitere Perspektiven wie z. B. eine Java-Perspektive mit Editoren und Ansichten für die Entwicklung von Java-Anwendungen.

Siehe „Flash Builder-Perspektiven und -Ansichten“ auf Seite 7.

Editor Mit einem Editor können Sie verschiedene Dateitypen bearbeiten. Welche Editoren zur Verfügung stehen, hängt von der Anzahl und Art der installierten Eclipse-Plug-Ins ab. Flash Builder enthält Editoren für MXML-, ActionScript 3.0- und CSS-(Cascading Style Sheets-)Code.

Weitere Informationen finden Sie unter „Editoren in Flash Builder“ auf Seite 21 und „Tools zur Codeentwicklung in Flash Builder“ auf Seite 49.

Ansichten Eine Ansicht unterstützt typischerweise einen Editor. Beispiel: Beim Bearbeiten von MXML werden in der Flash-Entwicklungsperspektive auch die Ansichten „Komponenten“ und „Flex-Eigenschaften“ angezeigt. Da diese Ansichten die Entwicklung von Anwendungen unterstützen, werden sie angezeigt, sobald eine MXML-Datei zur Bearbeitung geöffnet wird.

Weitere Informationen finden Sie unter „Mit Ansichten arbeiten“ auf Seite 20.

Ansicht ist ein Synonym für den Begriff *Bedienfeld*, der in früheren Versionen von Flash Builder, Adobe Dreamweaver® und anderen Entwicklungsprogrammen von Adobe verwendet wurde.

Arbeitsbereich Ein *Arbeitsbereich* (nicht mit *Workbench* zu verwechseln), ist ein definierter Bereich des Dateisystems, der die Ressourcen (Dateien und Ordner) enthält, aus denen Anwendungsprojekte bestehen. Sie können jeweils nur mit einem Arbeitsbereich arbeiten, haben aber die Möglichkeit, bei jedem Start von Flash Builder einen anderen Arbeitsbereich auszuwählen.

Weitere Informationen finden Sie unter „Projekte in Flash Builder“ auf Seite 35.

Ressource Der Begriff *Ressource* wird allgemein zur Bezeichnung der Dateien und Ordner in den Projekten in einem Arbeitsbereich verwendet.

Siehe „Projektressourcen“ auf Seite 42.

Projekt Alle Ressourcen, aus denen Anwendungen bestehen, befinden sich in Projekten. In Flash Builder werden Anwendungen in Projekten erstellt. Sie müssen also zunächst ein Projekt erstellen, bevor Sie Ihre Anwendung entwickeln können. Je nachdem, welchen Anwendungstyp Sie erstellen, unterstützt Flash Builder die verschiedenen Projekttypen.

Siehe „Projekttypen“ auf Seite 35 und „Projekte in Flash Builder“ auf Seite 90.

Startkonfiguration Das Programm erstellt für jedes Projekt eine *Startkonfiguration*. In dieser Startkonfiguration sind die Projekteinstellungen definiert, die beim Ausführen und Debuggen der Anwendungen verwendet werden. In der Startkonfiguration sind beispielsweise die Namen und Pfade der kompilierten SWF-Dateien der Anwendung enthalten. Sie können diese Einstellungen bei Bedarf ändern.

Siehe „Startkonfigurationen verwalten“ auf Seite 123.

Verwandte Themen

„Workbench anpassen“ auf Seite 266

Flash Builder-Perspektiven und -Ansichten

Um eine bestimmte Aufgabe oder eine Gruppe von Aufgaben zu unterstützen, sind Editoren und dazugehörige Ansichten in einer *Perspektive* kombiniert. Wenn Sie eine Datei öffnen, die mit einer bestimmten Perspektive verbunden ist, öffnet Flash Builder diese Perspektive automatisch.

Die eigenständige Konfiguration von Flash Builder enthält drei Perspektiven:

- Flash

  Siehe „Flash-Entwicklungsperspektive“ auf Seite 8.

- Flash-Debugger

  Siehe „Flash-Debugperspektive“ auf Seite 138.

- Flash-Profil

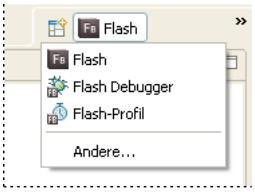
  Siehe „Flash-Profilingperspektive“ auf Seite 149.

  Hinweis: *Die Flash-Profil-Perspektive ist in Flash Builder Premium verfügbar.*

Perspektiven öffnen und wechseln

Perspektiven ändern sich automatisch, um die jeweilige Aufgabe zu unterstützen. Wenn Sie zum Beispiel ein Flex-Projekt erstellen, wechselt die Workbench zur Entwicklungsperspektive; wenn Sie eine Debugsitzung starten, wird die Flash-Debugperspektive angezeigt, sobald der erste Haltepunkt erreicht wird.

Sie können auch manuell zwischen Perspektiven wechseln, indem Sie „Fenster“ > „Perspektive öffnen“ wählen. Alternativ können Sie die gewünschte Perspektive aus der *Perspektivenleiste* in der Hauptsymbolleiste der Workbench auswählen.



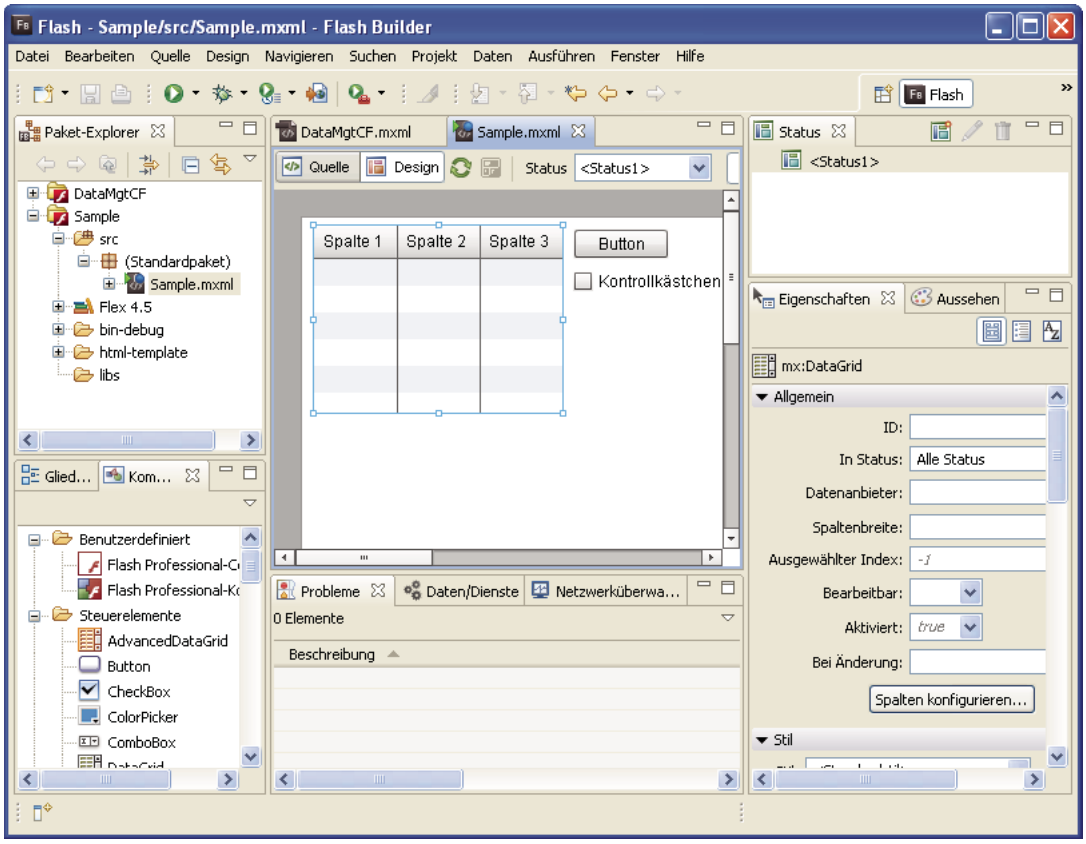
Wenn die Perspektive geöffnet wird, ändert sich ihr Titel und es wird der Name der von Ihnen ausgewählten Perspektive angezeigt. Neben dem Titel erscheint ein Symbol, über das Sie schnell zwischen Perspektiven im selben Fenster wechseln können. Perspektiven werden standardmäßig im selben Fenster geöffnet.

Wenn Sie die Plug-In-Konfiguration von Flash Builder verwenden und andere Eclipse-Plug-Ins installiert sind, stehen Ihnen u. U. zusätzliche Perspektiven zur Verfügung. Jedes Eclipse-Plug-In enthält vordefinierte Perspektiven, die Sie entsprechend Ihren Anforderungen anpassen können. Sie können auch eigene Perspektiven erstellen. Eine vollständige Liste der Perspektiven erhalten Sie über „Fenster“ > „Perspektive öffnen“ > „Andere“.

Flash-Entwicklungsperspektive

Die Flash-Entwicklungsperspektive enthält die Editoren und Ansichten, die zum Erstellen von Anwendungen für das Flex-Framework benötigt werden. Wenn Sie ein Projekt erstellen, wechselt Flash Builder zur Entwicklungsperspektive, damit Sie mit der Entwicklung der Anwendung beginnen können.

Das folgende Beispiel zeigt den Paket-Explorer, die Gliederungs- und die Problemansicht:



Der Fokusbereich der Perspektive (und der Workbench generell) ist der Editorbereich. Der Editorbereich enthält alle momentan geöffneten Dokumente auf einer aus mehreren Registerkarten bestehenden Benutzeroberfläche.

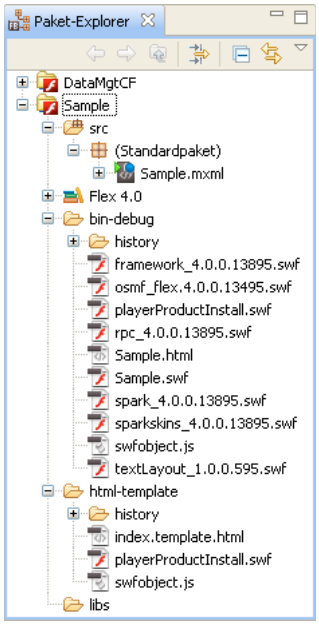
Wenn Sie ein Flex-Projekt erstellen, wird die MXML-Hauptanwendungsdatei im Editorbereich geöffnet. Sie können dann alle Dokumente (MXML, ActionScript und CSS), mit denen Sie arbeiten, öffnen und zwischen diesen Dokumenten wechseln. Weitere Informationen finden Sie unter „Editoren in Flash Builder“ auf Seite 21.

Flash-Entwicklungsperspektive im Quellmodus

Im Quellmodus (Codebearbeitung) besteht die Entwicklungsperspektive aus den Editoren und den folgenden Hilfsansichten:

Paket-Explorer-Ansicht

Der Paket-Explorer enthält alle Projekte und Ressourcen im Arbeitsbereich und ist eine der Hauptkomponenten der Flash Builder-Workbench. Der Paket-Explorer wird in der Entwicklungs- und Debugperspektive angezeigt.



Wenn Sie im Paket-Explorer arbeiten, können Sie eine Ressource auswählen und ihre Eigenschaften anzeigen.

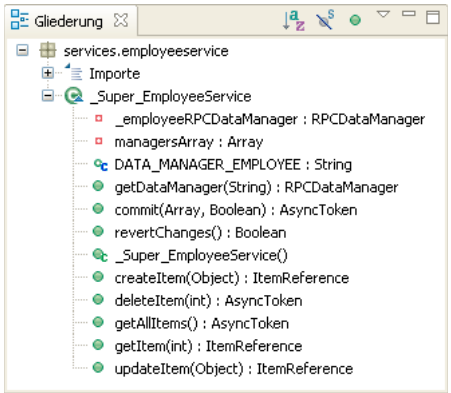
1 Wählen Sie im Paket-Explorer eine Ressource aus.

2 Wählen Sie „Datei“ > „Eigenschaften“.

Weitere Informationen zum Paket-Explorer und zum Arbeiten mit Projekten finden Sie unter „Projekte verwalten“ auf Seite 37.

Gliederungsansicht

Im Quellmodus zeigt die Gliederungsansicht eine hierarchische Ansicht der Codestruktur des ausgewählten MXML- oder ActionScript-Dokuments an, sodass Sie die Codeabschnitte oder -zeilen im Dokument inspizieren und zu ihnen navigieren können. Die Gliederungsansicht zeigt auch die vom Compiler generierten Syntaxfehlermeldungen an. Diese Ansicht ist auch verfügbar, wenn Sie den ActionScript-Editor verwenden.

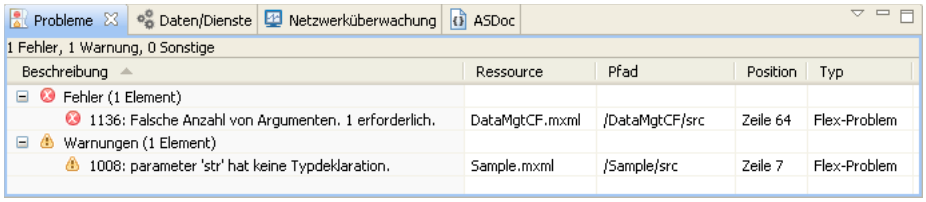


Weitere Informationen zum Verwenden der Gliederungsansicht im Quellmodus finden Sie unter „Gliederungsansicht verwenden“ auf Seite 81.

Problemansicht

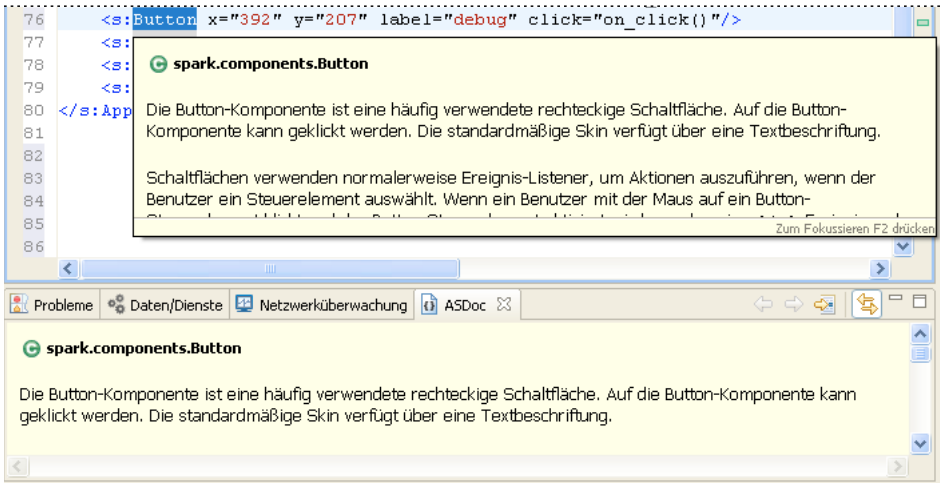
Wenn Sie Code eingeben, erkennt der Flash Builder-Compiler Syntaxfehler und andere Kompilierungsfehler und zeigt diese in der Problemansicht an. Im Paket-Explorer werden Knoten markiert, die Fehler enthalten.

Beim Debuggen von Anwendungen werden in der Problemansicht Fehler, Warnungen und andere Informationen angezeigt. Jeder Fehler bzw. jede Warnung enthält eine Meldung, die Datei und den Ordner, in dem sich der Fehler bzw. die Warnung befindet, sowie die Zeilennummer in der Datei. Probleme werden so lange in der Anzeige „Probleme“ angezeigt, bis Sie sie korrigieren oder sie auf sonstige Weise lösen.



Hinweis: *Sie können optional auch die Task- und Lesezeichenansicht hinzufügen. Diese Ansichten bieten zusätzliche Tastaturbefehle für die Verwaltung und Navigation im Code. Weitere Informationen zu diesen Ansichten finden Sie unter „Markierungen verwenden“ auf Seite 86. Eine Einführung in die optionalen Ansichten, die in Flash Builder verfügbar sind, finden Sie unter „Weitere nützliche Workbench-Ansichten“ auf Seite 18.*

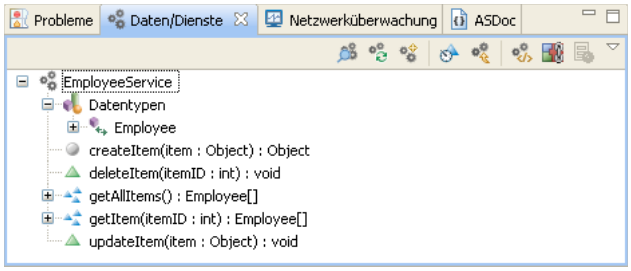
ASDoc-Ansicht


<s:Button x="392" y="207" label="debug" click="on_click()"/>
spark.components.Button
Die Button-Komponente ist eine häufig verwendete rechteckige Schaltfläche. Auf die Button-Komponente kann geklickt werden. Die standardmäßige Skin verfügt über eine Textbeschriftung.
Schaltflächen verwenden normalerweise Ereignis-Listener, um Aktionen auszuführen, wenn der Benutzer ein Steuerelement auswählt. Wenn ein Benutzer mit der Maus auf ein Button-
Zum Fokussieren F2 drücken
Probleme
Daten/Dienste
Netzwerküberwachung
ASDoc
spark.components.Button
Die Button-Komponente ist eine häufig verwendete rechteckige Schaltfläche. Auf die Button-Komponente kann geklickt werden. Die standardmäßige Skin verfügt über eine Textbeschriftung.

Flash Builder zeigt den für den jeweiligen Code relevanten ASDoc-Inhalt an, während Sie Code eingeben bzw. wenn Sie mit der Maus auf Codeelemente zeigen. Flash Builder zeigt außerdem in der ASDoc-Ansicht den ASDoc-Inhalt für den ausgewählten Code an.

Weitere Informationen finden Sie unter „ActionScript-Referenzübersicht anzeigen“ auf Seite 52.

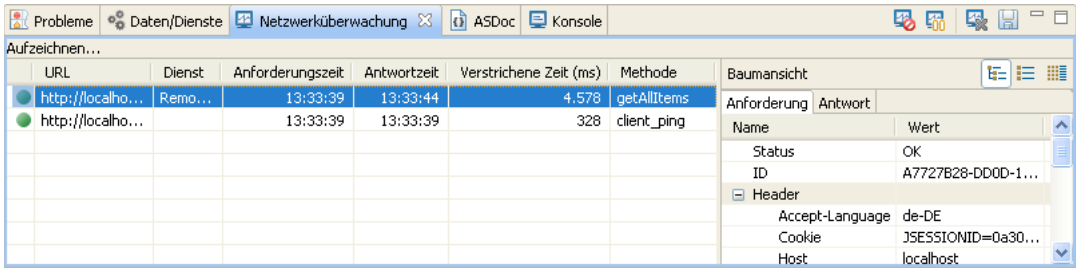
Ansicht „Daten/Dienste“



Flash Builder enthält Assistenten und Werkzeuge, mit denen Verbindungen zu Datendiensten hergestellt werden können. Verwenden Sie die Ansicht „Daten/Dienste“, um eine Verbindung zu Datendiensten herzustellen. Sobald die Verbindung zu einem Datendienst hergestellt wurde, zeigt die Ansicht „Daten/Dienste“ den Dienst, Datentypen für zurückgegebene Daten und verfügbare Methoden für den Dienst an. Verwenden Sie diese Ansicht, um den Zugriff auf den Dienst zu konfigurieren und um zurückgegebene Daten an Komponenten der Benutzeroberfläche zu binden.

Weitere Informationen hierzu finden Sie unter Building data-centric applications with Flash Builder.

Netzwerküberwachungsansicht

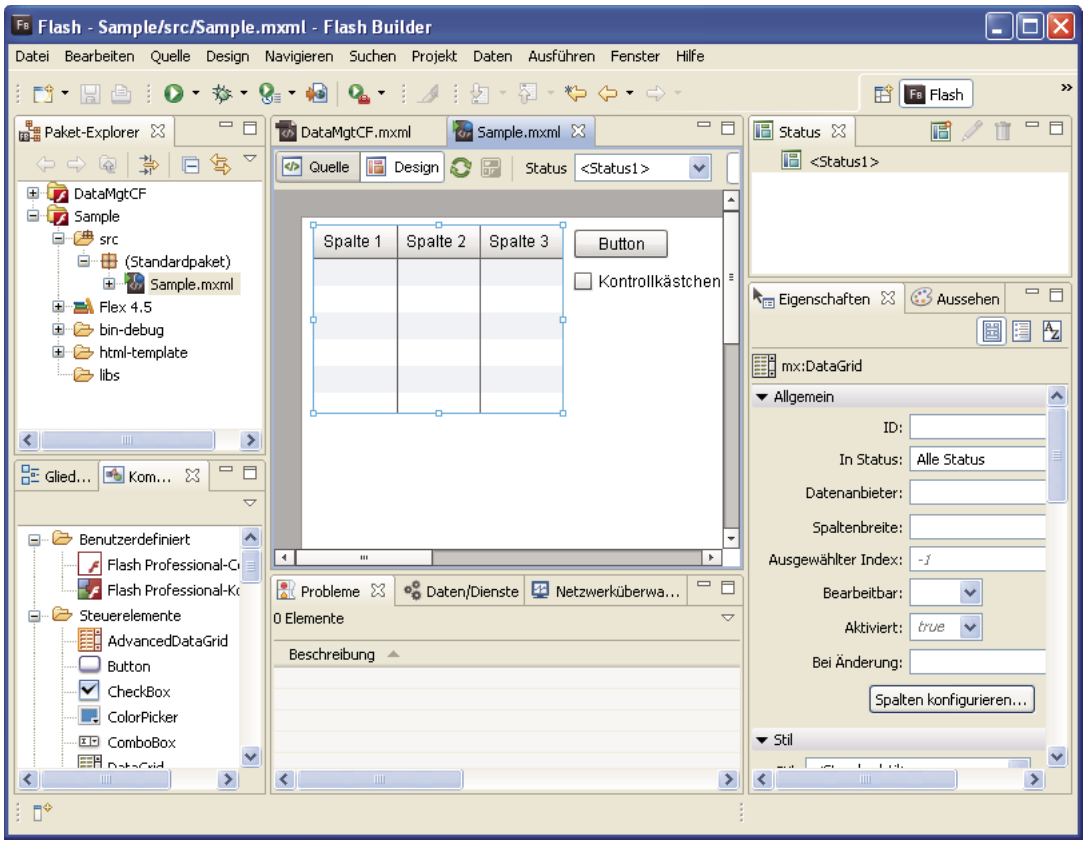


Mit der Netzwerküberwachung können Sie Daten überprüfen, die zwischen einer Anwendung und dem bzw. den Datendiensten ausgetauscht werden. Die Netzwerküberwachung ist in Flash Builder Premium verfügbar.

Weitere Informationen hierzu finden Sie unter „Anwendungen überwachen, die auf Datendienste zugreifen“ auf Seite 230.

Flash-Entwicklungsperspektive im Designmodus

Im Designmodus des MXML-Editors wird der Code, den Sie im Quellmodus bearbeiten, visuell dargestellt. Im Designmodus stehen zusätzliche Ansichten für Designaufgaben zur Verfügung, Hierbei handelt es sich um die folgenden Ansichten: Komponenten, Eigenschaften, Aussehen, Status und Designmodus-Probleme.



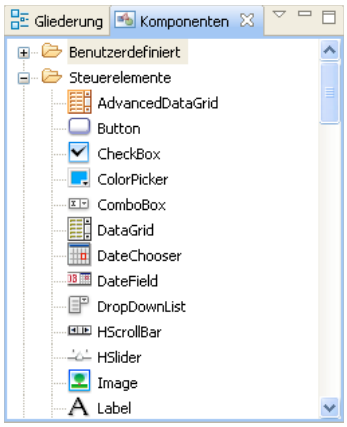
Außerdem wird im Designmodus in der Gliederungsansicht die MXML-Struktur der Anwendung angezeigt. Weitere Informationen zum Entwerfen von Anwendungen finden Sie unter „Benutzeroberflächen erstellen“ auf Seite 186.

Hinweis: *Der Designmodus ist nicht verfügbar, wenn Sie mit ActionScript-Projekten arbeiten. Um die Benutzeroberfläche von ActionScript-Anwendungen in der Vorschau anzuzeigen, erstellen Sie die Anwendungen und führen Sie sie aus. Weitere Informationen zum Umgang mit ActionScript finden Sie unter „ActionScript-Projekte“ auf Seite 36 und „Anwendungen ausführen und debuggen“ auf Seite 123.*

Im Designmodus enthält die Entwicklungsperspektive den MXML-Editor und folgende Ansichten:

Komponentenansicht

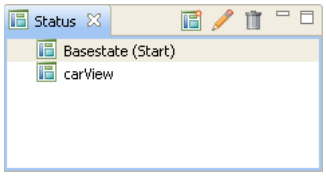
Die Komponentenansicht enthält alle Flex-Standardkomponenten, die Sie auswählen und zum Designbereich hinzufügen können. Wenn Sie eigene benutzerdefinierte Komponenten erstellen, werden diese ebenfalls in der Komponentenansicht angezeigt.



Weitere Informationen finden Sie unter „Flash-Komponenten erstellen und bearbeiten“ auf Seite 252.

Statusansicht

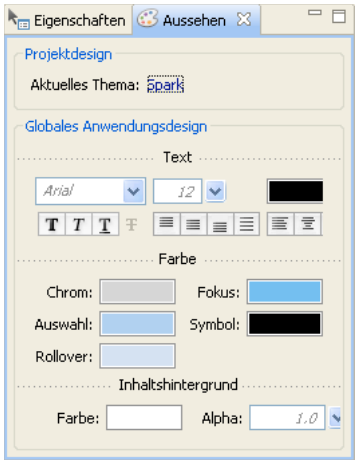
In Flex können Sie Anwendungen erstellen, deren Aussehen sich in Abhängigkeit von Ereignissen ändert, die entweder direkt vom Benutzer ausgelöst oder aber programmgesteuert generiert werden. Die Transformationen der Benutzeroberfläche werden als *Anzeigestatus* bezeichnet. Anzeigestatus werden in der Statusansicht erstellt und verwaltet.



Weitere Informationen zu Ansichtsstatus finden Sie unter „Anzeigestatus und Übergänge hinzufügen“ auf Seite 201.

Aussehen-Ansicht

In der Aussehen-Ansicht können Sie globale Anwendungsstile für Text und Farben festlegen. Sie können auch das Design für Anwendungen im Projekt angeben.

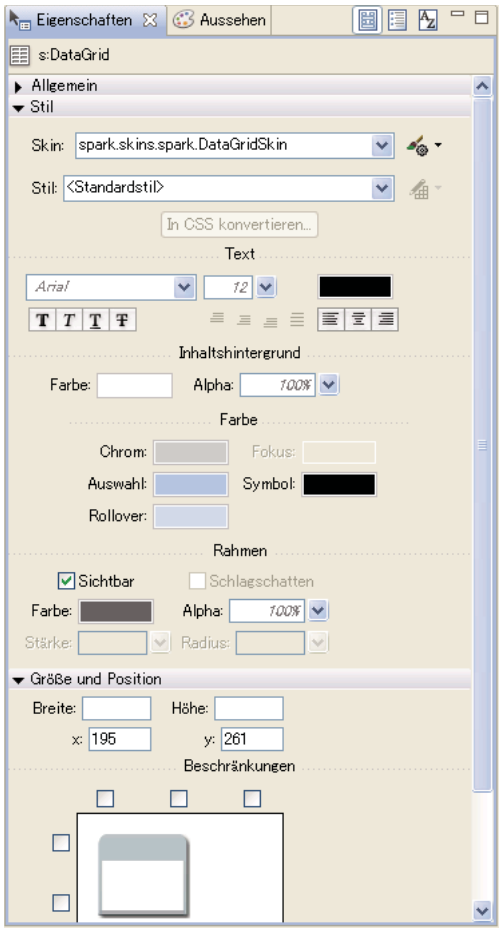


Aussehen-Bedienfeld

Weitere Informationen finden Sie unter „Stile auf eine Anwendung anwenden“ auf Seite 196.

Ansicht „Stil“

In der Ansicht „Stil“ legen Sie Stile für bestimmte Komponenten fest. Die Stile, die Sie festlegen können, variieren je nach Komponente.

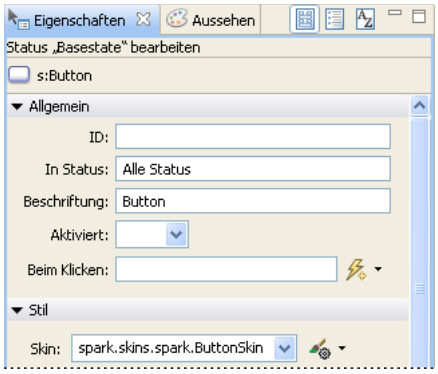


Stil-Bedienfeld

Weitere Informationen finden Sie unter „Stile auf eine Anwendung anwenden“ auf Seite 196.

Flex-Eigenschaftenansicht

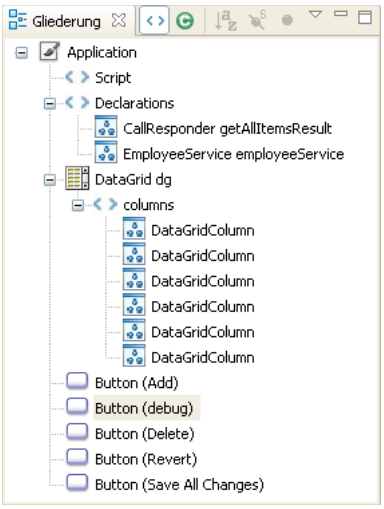
Wenn eine Flex-Komponente ausgewählt wird, werden ihre Eigenschaften in der Eigenschaftenansicht angezeigt. Sie können Eigenschaften wie gewünscht festlegen und bearbeiten. Eigenschaften einer Komponente können grafisch (wie im folgenden Beispiel gezeigt) und als kategorisierte oder alphabetische Liste angezeigt werden.



Weitere Informationen hierzu finden Sie unter „Komponenteneigenschaften festlegen“ auf Seite 28.

Gliederungsansicht

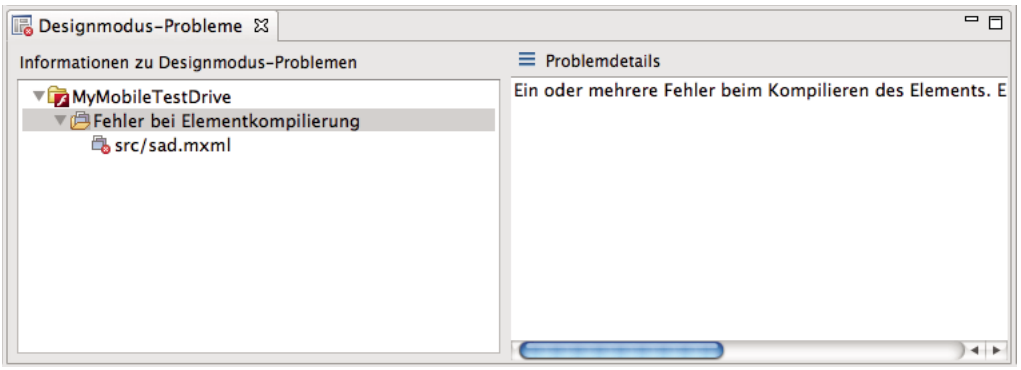
Im Designmodus zeigt die Gliederungsansicht eine hierarchische Ansicht der MXML-Struktur der Anwendungen an. Sie können bequem durch die Struktur einer Anwendung navigieren, indem Sie einzelne MXML-Tags und Komponenten auswählen. Wenn Sie ein Element in der Gliederungsansicht auswählen, wird es im Designmodus hervorgehoben.



Weitere Informationen zum Arbeiten mit der Gliederungsansicht im Designmodus finden Sie unter „Struktur einer MXML-Datei inspizieren“ auf Seite 32.

Ansicht „Designmodus-Probleme“

Bei Initialisierung der Designansicht werden möglicherweise erkannte Probleme im Zusammenhang mit dem Editor oder dem Projekt in der Ansicht „Designmodus-Probleme“ angezeigt.



Im Bedienfeld „Informationen zu Designmodus-Problemen“ wird die Liste von Problemen im Zusammenhang mit dem initialisierten Designmodus des Editors angezeigt. Die Anzeige der Probleme erfolgt im Baumstrukturformat.

Details zu einem Problem können im Bedienfeld „Informationen zu Designmodus-Problemen“ ausgewählt und dort angezeigt werden. Die im Zusammenhang mit diesem Projekt aufgetretenen Probleme werden beim Schließen des Projekts aus der Ansicht „Designmodus-Probleme“ entfernt.

Hinweis: *Die Ansicht „Designmodus-Probleme wird nur bei aktiviertem Designmodus angezeigt. Wenn Sie den Designmodus deaktivieren (deaktivieren Sie hierzu „Fenster“ >„Designmodus aktivieren“), wird die Ansicht „Designmodus-Probleme“ nicht angezeigt.*

Weitere nützliche Workbench-Ansichten

Neben den Editoren und Ansichten, die zu den standardmäßigen Entwicklungs-, Debug- und Profilperspektiven von Flash Builder gehören, bietet die Workbench auch andere Ansichten. Diese optionalen Ansichten sind nach Typen kategorisiert und mit spezifischen Funktionen in der Workbench oder mit spezifischen Eclipse-Plug-Ins verbunden.

Sie können Ansichten, die noch nicht mit einer Perspektive angezeigt werden, öffnen und in Workbench hinzufügen: Wählen Sie „Fenster“ > „Ansicht anzeigen“ > „Sonstiges“.

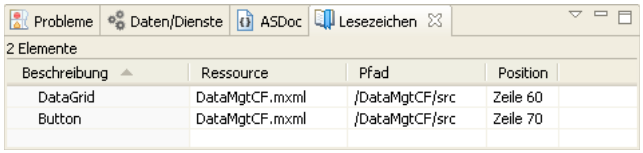
Die Ansichten „Tasks“, „Lesezeichen“ und „Suche“ unterstützen Sie bei der Anwendungsentwicklung.

Nachdem Sie eine Ansicht zur aktuellen Perspektive hinzugefügt haben, können Sie diese Ansicht als Teil der Perspektive speichern. Weitere Informationen hierzu finden Sie unter „Perspektiven anpassen“ auf Seite 19.

Sie können auch Schnellansichten erstellen, um häufig verwendete Ansichten schnell aufzurufen. Weitere Informationen hierzu finden Sie unter „Schnellansichten erstellen und anwenden“ auf Seite 21.

Lesezeichenansicht

Die Lesezeichenansicht dient zum Verwalten der Lesezeichen, die Sie zu spezifischen Codezeilen oder Ressourcen hinzufügen. Wie im Webbrowser werden über Lesezeichen Verknüpfungen zu interessanten Elementen erstellt, um sie schnell abrufen zu können. Wenn Sie ein Lesezeichen auswählen, wird es in der Workbench gesucht und angezeigt.

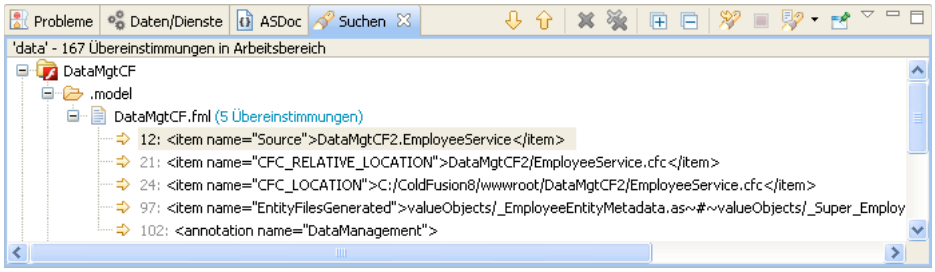


Verwandte Themen

„Markierungen verwenden“ auf Seite 86

Suchansicht

Die Suchansicht wird automatisch angezeigt, wenn Sie im Arbeitsbereich nach Ressourcen suchen. Sie können die Suchansicht verwenden, um Suchvorgänge zu definieren und frühere Suchvorgänge erneut aufzurufen, sowie um die Liste der Suchergebnisse zu filtern.



Verwandte Themen

„In der Workbench nach Ressourcen suchen“ auf Seite 44

Mit Perspektiven arbeiten

Standardperspektive festlegen

Das Wort *Standardwert*, das in Klammern auf den Namen der Perspektive folgt, bezeichnet die Standardperspektive.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Perspektiven“.

2 Wählen Sie unter „Verfügbare Perspektiven“ die Perspektive aus, die als Standardperspektive definiert werden soll, und klicken Sie auf „Als Standard definieren“.

3 Klicken Sie auf „OK“.

Perspektive in neuem Fenster öffnen

Sie können angeben, dass Perspektiven in einem neuen Fenster geöffnet werden sollen.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Perspektiven“.

2 Aktivieren Sie unter „Neue Perspektive öffnen“ die Option „In neuem Fenster“.

Um wieder die Standardeinstellung zu verwenden, aktivieren Sie die Option „In gleichem Fenster“.

3 Klicken Sie auf „OK“.

Perspektiven anpassen

Um das Layout einer Perspektive anzupassen, ändern Sie die Editoren und Ansichten, die in der jeweiligen Perspektive sichtbar sind. Beispielsweise könnten Sie die Lesezeichenansicht in einer Perspektive eingeblendet und in einer anderen Perspektive ausgeblendet haben.

So erstellen Sie eine Perspektive:

1 Öffnen Sie eine vorhandene Perspektive.

2 Rufen Sie die gewünschten Ansichten und Editoren auf.

3 Wählen Sie „Fenster“ > „Perspektive“ > „Perspektive speichern unter“ („Fenster“ > „Perspektive speichern unter“ in der Plug-In-Konfiguration von Flash Builder).

4 Geben Sie im Dialogfeld „Perspektive speichern unter“ einen neuen Namen für die Perspektive ein und klicken Sie dann auf „OK“.

So konfigurieren Sie eine Perspektive:

1 Öffnen Sie die Perspektive, die Sie konfigurieren möchten.

2 Wählen Sie „Fenster“ > „Perspektive anpassen“.

3 Klicken Sie je nach den Elementen, die Sie zur angepassten Perspektive hinzufügen möchten, auf die Registerkarte „Schnellzugriff“ oder „Befehle“.

4 Verwenden Sie die Kontrollkästchen, um auszuwählen, welche Elemente in Menüs und Symbolleisten in der ausgewählten Perspektive sichtbar sein sollen.

5 Klicken Sie auf „OK“.

6 Wählen Sie „Fenster“ > „Perspektive speichern als“.

7 Geben Sie im Dialogfeld „Perspektive speichern unter“ einen neuen Namen für die Perspektive ein und klicken Sie auf „OK“.

Wenn Sie eine Perspektive speichern, fügt Flash Builder den Namen der neuen Perspektive dem Menü „Fenster“ > „Perspektive öffnen“ hinzu.

Angepasste Perspektiven löschen

Sie können Perspektiven löschen, die zuvor definiert wurden. Sie können keine Perspektiven löschen, die nicht von Ihnen erstellt wurden.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Perspektiven“.

2 Wählen Sie unter „Verfügbare Perspektiven“ die Perspektive aus, die Sie löschen möchten.

3 Klicken Sie auf „Löschen“ und dann auf „OK“.

Perspektive zurücksetzen

Sie können eine Perspektive, die Sie geändert haben, wieder auf ihr Originallayout zurücksetzen.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Perspektiven“.

2 Wählen Sie unter „Verfügbare Perspektiven“ die Perspektive aus, die zurückgesetzt werden soll.

3 Klicken Sie auf „Zurücksetzen“ und dann auf „OK“.

Mit Ansichten arbeiten

Ansichten verschieben und andocken

Sie können Ansichten an verschiedene Positionen in der Workbench verschieben und sie bei Bedarf an- oder abdocken.

1 Ziehen Sie die Ansicht an ihrer Titelleiste an die gewünschte Position.

Wenn Sie die Ansicht in der Workbench bewegen, ändert sich der Zeiger in einen Ablagecursor. Der Ablagecursor zeigt an, wo Sie die Ansicht andocken, wenn Sie die Maustaste loslassen.

Sie können eine Gruppe gestapelter Ansichten ziehen, indem Sie vom leeren Bereich rechts neben den Registerkarten der Ansichten aus ziehen.

Sie können eine Ansicht auch über das Kontextmenü der Ansicht verschieben. Öffnen Sie das Kontextmenü über die Registerkarte der Ansicht, wählen Sie „Verschieben“ > „Ansicht“ und verschieben Sie die Ansicht an die gewünschte Position. Klicken Sie erneut mit der Maustaste.

2 (Optional) Speichern Sie Ihre Änderungen mithilfe der Option „Fenster“ > „Perspektive speichern als“.

Reihenfolge von Ansichten im Registerkartenformat ändern

Außer die Ansichten an verschiedenen Positionen in der Workbench anzudocken, können Sie auch die Reihenfolge der Ansichten in einer Registerkartengruppe ändern.

❖ Klicken Sie auf die Registerkarte der Ansicht, die verschoben werden soll, ziehen Sie die Ansicht an die gewünschte Position und lassen Sie die Maustaste los. Wenn Sie die Ansicht über Registerkarten anderer Ansichten ziehen, wird ein Stapelsymbol angezeigt.

Zwischen Ansichten wechseln

Um zwischen Ansichten zu wechseln, haben Sie die folgenden Möglichkeiten:

- Klicken Sie auf die Registerkarte einer anderen Ansicht.
- Wählen Sie im Menü „Fenster“ von Flash Builder eine Ansicht.
- Verwenden Sie einen Tastaturbefehl.

Drücken Sie auf Windows-Plattformen Strg+F7 und auf Macintosh-Plattformen Befehl+F7. Drücken Sie F7, um eine Ansicht zu wählen.

Eine Liste aller Tastaturbefehle finden Sie unter „Hilfe“ > „Hilfe für Tastenbelegung“.

Schnellansichten erstellen und anwenden

Schnellansichten sind ausgeblendete Ansichten, die Sie schnell öffnen und schließen können. Sie funktionieren wie andere Ansichten, nehmen aber keinen Platz in der Workbench weg, während Sie arbeiten.

Die Ansicht wird geöffnet, wenn Sie auf das Schnellansichtsymbol in der Schnellzugriffsleiste klicken. Sobald Sie auf eine beliebige Stelle außerhalb der Schnellansicht (oder in der Schnellansichtsymbolleiste auf „Minimieren“) klicken, wird die Ansicht wieder ausgeblendet.

Hinweis: *Wenn Sie die Paket-Explorer-Ansicht in eine Schnellansicht umwandeln und dann eine Datei über die Schnellansicht des Paket-Explorers öffnen, wird die Schnellansicht automatisch ausgeblendet und Sie können mit der jeweiligen Datei arbeiten.*

So erstellen Sie eine Schnellansicht:

- Ziehen Sie die Ansicht, aus der Sie eine Schnellansicht machen möchten, in die Schnellzugriffsleiste unten links im Workbench-Fenster.

  Das Symbol für die Ansicht, die Sie gezogen haben, wird in der Schnellzugriffsleiste angezeigt. Sie können die Ansicht öffnen, indem Sie auf ihr Symbol in der Schnellzugriffsleiste klicken. Sobald Sie auf eine Stelle außerhalb der Ansicht klicken, wird die Ansicht wieder ausgeblendet.

So ändern Sie eine Schnellansicht in eine Normalansicht:

- Deaktivieren Sie im Kontextmenü der Ansicht den Befehl „Schnellansicht“.

Task- und Problemansicht filtern

Sie können die Tasks bzw. Fehler, die in der Task- und Problemansicht angezeigt werden, filtern. Wenn Sie z. B. nur Fehler anzeigen möchten, die die Workbench protokolliert hat, oder nur Tasks, die Sie zur eigenen Erinnerung protokolliert haben, dann filtern Sie nach der Ressource oder Ressourcengruppe, mit denen sie verknüpft sind. Es ist möglich, nach Textstring im Feld „Beschreibung“, Schweregrad des Fehlers, Taskpriorität und Taskstatus zu filtern.

1. Klicken Sie in der Taskleiste der Task- oder Problemansicht auf „Filtern“.
2. Geben Sie die entsprechenden Informationen im Dialogfeld „Filter“ ein und klicken Sie auf „OK“.

Weitere Informationen zum Arbeiten mit Ansichten finden Sie unter „Mit Ansichten arbeiten“ auf Seite 20.

Editoren in Flash Builder

Wenn Sie in Flash Builder Anwendungen entwickeln, verwenden Sie hierzu den MXML-, ActionScript 3.0- und CSS-Editor.

Editoren sind mit Ressourcentypen verbunden. Wenn Sie Ressourcen in der Workbench öffnen, wird hierdurch also automatisch der entsprechende Editor geöffnet. Jeder Editor stellt die für den jeweiligen Ressourcentyp erforderliche Funktionalität zur Verfügung.

MXML-Editor Der MXML-Editor wird zum Bearbeiten von MXML- und zum Einbetten von ActionScript- und CSS-Code in `<fx:Script>`- und `<fx:Style>`-Tags verwendet. Der MXML-Editor hat zwei Modi: Quelle und Design. Der Quellmodus dient zur Bearbeitung von Code. Der Designmodus dient zum visuellen Layouten und Gestalten von

Anwendungen. Die beiden Modi sind synchronisiert. Änderungen in einem Modus sind also sofort im anderen Modus sichtbar.

ActionScript-Editor Der ActionScript-Editor dient zum Bearbeiten von ActionScript-Klassen- und -Schnittstellendateien. Sie können ActionScript-Funktionen in eine MXML-Datei einbetten, indem Sie das `<fx:Script>`-Tag verwenden. Es ist aber üblicher, Klassen in separaten ActionScript-Dateien zu definieren und die Klassen dann in MXML-Dateien zu importieren. Mit dieser Methode können Sie die meisten Anwendungen in ActionScript definieren.

CSS-Editor Der CSS-Editor dient zum Anzeigen und Bearbeiten von Cascading Style Sheets. Sie können anschließend auf die visuellen Elemente der Anwendungen Stile anwenden. Weitere Informationen hierzu finden Sie unter Stile und Designs.

Hinweis: *Der CSS-Editor eignet sich nur für Projekte, die das Flex 4 SDK verwenden. Bei Projekten, die das Flex 3 SDK verwenden, arbeitet der CSS-Editor in zwei Modi (Quelle und Design).*

Verwandte Themen

„Tools zur Codeentwicklung in Flash Builder“ auf Seite 49

Tipps und Tricks für Flash Builder

MXML-Editor im Designmodus verwenden

MXML-Datei im Designmodus anzeigen

1 Wenn die MXML-Datei noch nicht im MXML-Editor geöffnet ist, doppelklicken Sie auf die Datei im Paket-Explorer, um sie zu öffnen.

2 Wenn der MXML-Editor Quellcode anzeigt, klicken Sie am oberen Rand des Editorbereichs auf „Design“.

Sie können schnell zwischen Modi wechseln, indem Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) + # drücken.

Wenn Sie zwischen Quell- und Designmodus wechseln, werden automatisch designbezogene Ansichten wie die Komponenten-, Eigenschaften- und Statusansicht ein- bzw. ausgeblendet. Um dieses Verhalten zu aktivieren oder zu deaktivieren, wählen Sie „Fenster“ > „Voreinstellungen“ aus, klicken auf „Flash Builder“ > „Editoren“ > „Designmodus“ und aktivieren bzw. deaktivieren dann die Option „Mit dem Design verknüpfte Ansichten automatisch anzeigen“.

Horizontaler und vertikaler Bildlauf im Designbereich

❖ Klicken Sie auf der rechten Seite der Symbolleiste des Editors auf die Schaltfläche „Schwenkmodus“. Drücken Sie die Taste „H“ auf der Tastatur, um den Schwenkmodus aufzurufen. Um vorübergehend zum Schwenkmodus zu wechseln, drücken Sie die Leertaste auf der Tastatur und halten Sie sie gedrückt. Im Schwenkmodus können Sie keine Elemente auswählen bzw. verschieben.

In den Designbereich einzoomen

Das Zoomwerkzeug kann auf mehrere Weisen verwendet werden. Sie können Prozentwerte aus dem Haupt- oder dem Popupmenü auswählen, in der Symbolleiste auf die Schaltfläche „Zoommodus“ klicken oder Tastaturbefehle verwenden. In der Symbolleiste wird immer der aktuelle Zoomfaktor angezeigt.

• Wählen Sie im Hauptmenü „Design“ > „Einzoomen“ bzw. „Design“ > „Auszoomen“. Sie können auch das Untermenü „Vergrößerung“ auswählen und einen bestimmten Prozentsatz wählen.

- Klicken Sie in der Symbolleiste auf die Schaltfläche „Zoommodus“ oder drücken Sie die Taste „Z“ auf der Tastatur. Im Designbereich erscheint ein Cursor mit einem Plussymbol.
- Wählen Sie in der Symbolleiste des Editors aus dem Popupmenü neben den Schaltflächen „Auswahlmodus“, „Schwenkmodus“ und „Zoommodus“ einen Prozentwert aus. Die Größe des Designbereichs ändert sich entsprechend dem ausgewählten Prozentwert bzw. der Designbereich wird in das Fenster eingepasst.
- Klicken Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) in den Designbereich und wählen Sie „Einzoomen“, „Auszoomen“ oder das Untermenü „Vergrößerung“ aus. Der Designbereich ändert sich entsprechend Ihrer Auswahl.

Wenn Sie sich im Designbereich befinden, können Sie immer die folgenden Tastaturbefehle verwenden:

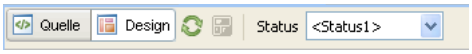
- Einzoomen: Strg+= (Mac OS: Befehl+=)
- Auszoomen: Strg+- (Mac OS: Befehl+-)

Weitere Tastaturbefehle finden Sie unter „Hilfe“ > „Hilfe für Tastenbelegung“.

Designmodus zur korrekten Anzeige aktualisieren

Sie können den Designmodus des MXML- und CSS-Editors bei Bedarf aktualisieren, damit das Layout richtig angezeigt wird. Die Layoutanzeige kann in bestimmten Situationen veraltet sein. Dies kann zum Beispiel vorkommen, wenn Sie ein visuelles Element in einer abhängigen Flash-Komponente (SWC) ändern. Stile und Skins werden manchmal nicht richtig angezeigt, weil Flash Builder die Datei neu erstellen muss.

❖ Klicken Sie in der Symbolleiste des Editors auf die Schaltfläche „Aktualisieren“.



Im Quell- und Designmodus gleichzeitig arbeiten

Sie können auch eine geteilte Ansicht erstellen, damit Sie simultan im Quell- und Designmodus arbeiten können.

1 Wählen Sie im Optionsmenü der Registerkarte des Editors die Option „Neuer Editor“.

Sie haben jetzt zwei Editorregisterkarten für die gleiche Datei.

2 Ziehen Sie eine der Registerkarten nach rechts, um die Editorfenster nebeneinander zu stellen.

3 Setzen Sie einen der Editoren auf „Designmodus“ und den anderen auf „Quellmodus“.

Um nur im Quellmodus zu arbeiten, deaktivieren Sie im Menü „Fenster“ die Option „Designmodus aktivieren“.

Visuell mit Komponenten im MXML-Designmodus arbeiten

Im Designmodus gestattet der MXML-Editor eine visuelle Bearbeitung der Komponenten. So können Sie sehen, wie Ihre Anwendung aussehen wird. Sie können Elemente im Designbereich ansehen, auswählen, scrollen, Elemente verschieben, skalieren und vergrößern. Dies ist besonders bei eingebetteten Containerkomponenten nützlich.

Komponentenansicht

Im Designmodus des MXML-Editors zeigt Flash Builder die Komponentenansicht an, aus der Sie beim Erstellen des Layouts Ihrer Anwendung Komponenten in den Designbereich ziehen.

Standardmäßig zeigt die Komponentenansicht die für ein Projekt empfohlenen Komponenten an. Die Version des Flex SDK für ein Projekt ist dafür ausschlaggebend, welche Komponenten als empfohlene Komponenten angezeigt werden.

Wenn Sie beispielsweise ein Projekt erstellen, verwendet Flash Builder standardmäßig das Flex 4.6 SDK. Für das Flex 4.6-SDK werden die Spark-Komponenten empfohlen. MX-Komponenten, für die es keine entsprechende Spark-Komponente gibt, werden ebenfalls als empfohlene Komponenten angezeigt.

Die Komponentenanzeige fasst Komponenten nach Kategorie zusammen, z. B. Steuerelemente, Layout, Navigatoren und Diagramme. Ferner gibt es eine Kategorie „Benutzerdefiniert“, die Komponenten auflistet, die in separaten MXML- und ActionScript-Dateien definiert sind. Weitere Informationen finden Sie unter „Benutzerdefinierte MXML-Komponenten erstellen“ auf Seite 239.

Hinweis: *Die Komponentenansicht listet sichtbare Komponenten auf.*

Darstellung von Komponenten in der Komponentenansicht modifizieren

Verwenden Sie das Ansichtsmenü in der Komponentenansicht, um festzulegen, wie Komponenten angezeigt werden.

1 Deaktivieren Sie die Option „Nur empfohlene Komponenten anzeigen“, um alle Komponenten anzuzeigen.

Wenn Sie angeben, dass alle Komponenten angezeigt werden sollen, fasst die Komponentenansicht die Komponenten nach Spark- und MX-Komponenten zusammen.

2 Wenn Sie voll qualifizierte Klassennamen anzeigen möchten, aktivieren Sie die Option „Vollständig qualifizierte Klassennamen anzeigen“.

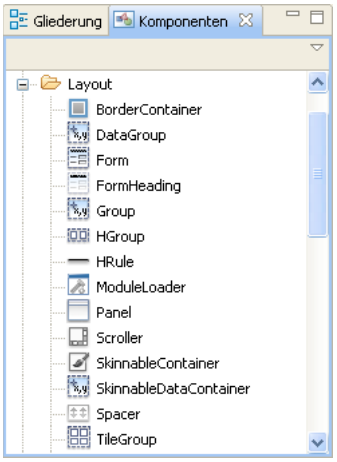
Komponenten im MXML-Designmodus hinzufügen

1 Öffnen Sie im Designmodus des MXML-Editors die MXML-Datei, in die Sie die Komponente einfügen möchten.

Um die Komponentenansicht zu verwenden, muss eine MXML-Datei im Designmodus geöffnet sein. Die MXML-Datei kann die Hauptanwendungsdatei (eine Datei mit einem Anwendungscontainer) oder eine benutzerdefinierte MXML-Kompomentendatei sein.

2 Wählen Sie in der Komponentenansicht die Komponente aus, die Sie hinzufügen möchten.

Wenn die Komponentenansicht nicht geöffnet ist, wählen Sie „Fenster“ > „Komponenten“ aus.



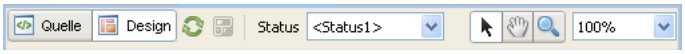
Die Komponentenansicht ordnet Komponenten nach Kategorie. Außerdem zeigt die Komponentenansicht empfohlene Komponenten für ein Projekt an. Sie können die Anzeigeeinstellungen ändern, um alle Komponenten aufzulisten. Weitere Informationen finden Sie unter „Komponentenansicht“ auf Seite 14.

3 Ziehen Sie eine Komponente aus der Komponentenansicht in die MXML-Datei.

Die Komponente wird im Layout entsprechend der Layoutregel des übergeordneten Containers platziert.

Komponenten im Designbereich auswählen und verschieben

❖ Klicken Sie auf der rechten Seite der Symbolleiste des Editors auf die Schaltfläche „Auswahlmodus“ (Pfeil). Wenn Sie ein Dokument öffnen, ist der Auswahlmodus standardmäßig aktiviert. Drücken Sie die Taste „V“ auf der Tastatur, um den Auswahlmodus aufzurufen. Ziehen Sie eine Komponente an die gewünschte Stelle. Sie können auch ziehen, um zu skalieren, und klicken, um auszuwählen.



Mehrere Komponenten aus- und abwählen

Sie können mehrere Komponenten in einer MXML-Datei auswählen. Dies kann nützlich sein, wenn Sie einen gleichen Wert für eine bestimmte Eigenschaft festlegen möchten.

- Klicken Sie bei gedrückter Strg-Taste (Mac OS: bei gedrückter Befehlstaste) auf jede Komponente im Layout.
- Klicken Sie auf den Hintergrund der Seite und ziehen Sie einen Rahmen, der die Komponenten überlappt.
- Klicken Sie in der Gliederungsansicht („Fenster“ > „Gliederung“) bei gedrückter Strg-Taste (Mac OS: bei gedrückter Befehlstaste) auf die Komponenten in der Ordnerstruktur.

Zum Abwählen mehrerer Komponenten verfahren Sie wie folgt:

- Klicken Sie auf den Hintergrund-Container.
- Klicken Sie auf eine nicht ausgewählte Komponente.
- Klicken Sie in den grauen Rand um die Stammkomponente.

Komponenten positionieren

Je nach der Layouteigenschaft des übergeordneten Containers können Sie die Position der Komponenten visuell ändern. Wenn der übergeordnete Container absolute Positionierung angibt, können Sie Komponenten per Drag & Drop neu platzieren. Ansonsten befolgen neu platzierte Komponenten die Layoutregeln des übergeordneten Containers.

Standardmäßig verwenden Spark-Container die BasicLayout-Eigenschaft, die absolute Positionierung erlaubt. Für manche MX-Container können Sie auch die Layout-Eigenschaft `absolute` angeben. Weitere Informationen finden Sie unter MX layout containers.

Hinweis: *Sie können auch Layoutbeschränkungen verwenden, um eine Komponente in einem Container zu platzieren. Layoutbeschränkungen geben an, wie Komponenten neu zu platzieren sind, wenn die Größe eines Containers verändert wird. Sie können jedoch auch verwendet werden, um die Position in Containern anzugeben, die eine feste Größe haben. Siehe Beschränkungen zum Steuern des Komponentenlayouts verwenden.*

1 Wählen Sie im Designmodus des MXML-Editors die gewünschte Komponente im Layout aus und ziehen Sie sie an eine neue Position.

   Die Komponente wird entsprechend der Layoutregeln des übergeordneten Containers im Layout platziert.

   Wenn der Container über absolute Positionierung verfügt, können Sie Komponenten an jede Stelle im Container ziehen und platzieren.

   Wenn Sie beispielsweise einen VGroup-Container in einen HGroup-Container verschieben, wird der VGroup-Container in die horizontale Anordnung mit den anderen untergeordneten Containern (sofern vorhanden) platziert.

2 Wählen Sie im Designmodus den übergeordneten Container der Komponente aus und bearbeiten Sie die Layouteigenschaften der Komponente in der Eigenschaftenansicht.

In einigen Fällen können Sie die Position untergeordneter Komponenten ändern, indem Sie die Eigenschaften des übergeordneten Containers ändern. Beispielsweise können Sie die Eigenschaften `verticalGap` und `horizontalGap` eines Containers verwenden, um den Abstand zwischen untergeordneten Komponenten festzulegen, und über die Eigenschaft `direction` ein Zeilen- oder Spaltenlayout definieren.

Größe der Komponenten anpassen

Sie können die Größe einer Komponente im Designmodus ändern, indem Sie mithilfe des Designmenüs einen Größenänderungsgriff ziehen oder ihre Eigenschaften in der Eigenschaftenansicht bearbeiten.

Hinweis: *Verwenden Sie ein beschränkungsbasiertes Layout, um anzugeben, wie die Größe einer Komponente dynamisch angepasst wird, wenn die Größe eines Containers geändert wird. Weitere Informationen finden Sie unter Beschränkungen zum Steuern des Komponentenlayouts verwenden.*

Größe einer Komponente visuell ändern

❖ Klicken Sie im Designmodus des MXML-Editors auf die Komponente und ziehen Sie einen Größenänderungsgriff, um die Größe der Komponente zu ändern.

- Um die Proportionen der Komponente beizubehalten, halten Sie beim Ziehen die Umschalttaste gedrückt.
- Wenn die automatische Ausrichtungsfunktion aktiviert ist, werden beim Ändern der Größe Ausrichtungslinien angezeigt, um die Ränder der Komponente an einer Linie mit benachbarten Komponenten auszurichten. Um die automatische Ausrichtungsfunktion über das Hauptmenü zu aktivieren bzw. zu deaktivieren, wählen Sie „Design“ > „Objektausrichtung aktivieren“ aus.

Gleiche Breite oder Höhe für zwei oder mehr Komponenten einstellen

1 Wählen Sie im Designmodus zwei oder mehr Komponenten aus.

2 Wählen Sie im Menü „Design“ eine der folgenden Skalierungsoptionen aus:

Gleiche Breite Setzt die Eigenschaft `Breite` für alle ausgewählten Komponenten auf die Breite der Komponente, die Sie zuerst ausgewählt haben.

Gleiche Höhe Setzt die Eigenschaft `Höhe` für alle ausgewählten Komponenten auf die Höhe der Komponente, die Sie zuerst ausgewählt haben.

Wenn sich alle ausgewählten Komponenten im gleichen Container befinden und für die erste Komponente, die Sie auswählen, eine Breite oder Höhe in Prozent angegeben ist, werden alle Elemente auf diese Breite oder Höhe in Prozent gesetzt. Andernfalls werden alle Komponenten auf die gleiche Pixelbreite bzw. -höhe gesetzt.

Größe einer Komponente durch Bearbeitung ihrer Eigenschaften ändern

1 Wählen Sie die Komponente im Designmodus aus.

Wenn Sie bei gedrückter Strg-Taste (unter Mac OS bei gedrückter Umschalttaste) auf mehrere Komponenten klicken, können Sie deren Größen gleichzeitig festlegen.

2 Legen Sie in der Eigenschaftenansicht („Fenster“ > „Eigenschaften“) die Eigenschaft `Höhe` bzw. `Breite` für die ausgewählten Komponenten fest.

Die Eigenschaftenansicht bietet drei Ansichten zur Inspektion der Eigenschaften einer Komponente: eine Standardansicht, eine Kategorieansicht und eine alphabetische Ansicht. Sie können zwischen diesen Ansichten wechseln, indem Sie auf die Ansichtsschaltflächen in der Symbolleiste klicken.

Automatische Ausrichtungsfunktion zur Positionierung von Komponenten verwenden

Wenn Sie eine Komponente visuell in einen Container ziehen, der über absolute Positionierung verfügt, wird die Komponente u. U. automatisch ausgerichtet. Die Ausrichtung erfolgt dabei relativ zu vorhandenen Komponenten. Die Komponenten können vertikal oder horizontal ausgerichtet werden.

Standardmäßig verwenden Spark-Container die BasicLayout-Eigenschaft, die absolute Positionierung erlaubt. Für manche MX-Container können Sie auch die Layout-Eigenschaft `absolute` angeben. Weitere Informationen finden Sie unter MX layout containers.

Sie können die automatische Ausrichtungsfunktion für eine oder alle Komponenten deaktivieren.

Automatische Ausrichtungsfunktion aktivieren oder deaktivieren

- Öffnen Sie die gewünschte MXML-Datei im Designmodus des MXML-Editors (sofern sie noch nicht geöffnet ist) und aktivieren (bzw. deaktivieren) Sie die Option „Design“ > „Objektausrichtung aktivieren“.

Automatische Ausrichtungsfunktion als Voreinstellung aktivieren oder deaktivieren

1. Öffnen Sie das Dialogfeld „Voreinstellungen“.
2. In der Seitenleiste des Dialogfensters „Voreinstellungen“ wählen Sie „Flash Builder“ > „Editoren“ > „Designmodus“.
3. Aktivieren bzw. deaktivieren Sie die Option „Objektausrichtung aktivieren“.

Komponenten ausrichten

Sie können Komponenten visuell relativ zueinander in einem Container ausrichten, der über absolute Positionierung verfügt. Standardmäßig verwenden Spark-Container die BasicLayout-Eigenschaft, die absolute Positionierung erlaubt. Für manche MX-Container können Sie auch eine Layout-Eigenschaft von `absolute` wählen.

Wenn Sie ein beschränkungsbasiertes Layout verwenden, können Sie Komponenten in einem Container auch zentrieren. Weitere Informationen finden Sie unter Beschränkungen zum Steuern des Komponentenlayouts verwenden.

Komponenten in einem Container ausrichten, der über absolute Positionierung verfügt

1. Wählen Sie im Designmodus des MXML-Editors zwei oder mehr Komponenten im Container aus.

   Weitere Informationen finden Sie unter „Mehrere Komponenten aus- und abwählen“ auf Seite 25.

2. Wählen Sie im Menü „Design“ eine der folgenden Ausrichtungsoptionen aus:

   Linksbündig Positioniert alle ausgewählten Komponenten so, dass ihre linken Ränder an den Rändern der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

   Vertikale Mitten ausrichten Platziert alle ausgewählten Komponenten so, dass ihre vertikalen Mitten an der vertikalen Mitte der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

   Rechtsbündig Positioniert alle ausgewählten Komponenten so, dass ihre rechten Ränder an den Rändern der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

   Oben ausrichten Positioniert alle ausgewählten Komponenten so, dass ihre oberen Ränder an den Rändern der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

   Horizontale Mitten ausrichten Platziert alle ausgewählten Komponenten so, dass ihre horizontalen Mitten an der horizontalen Mitte der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

   Unten ausrichten Positioniert alle ausgewählten Komponenten so, dass ihre unteren Ränder an den Rändern der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind.

Grundlinien ausrichten Platziert alle ausgewählten Komponenten so, dass ihre horizontalen Textgrundlinien an den Textgrundlinien der Komponente, die Sie zuerst ausgewählt haben, ausgerichtet sind. Bei Komponenten, die keine Textgrundlinie haben (wie z. B. HGroup), wird der untere Rand als Grundlinie betrachtet.

Bei Objekten ohne Layoutbeschränkung passt Flash Builder die Eigenschaft `x` bzw. `y` an, um die vertikale bzw. horizontale Ausrichtung zu ändern.

Bei Objekten mit Layoutbeschränkung passt Flash Builder die linke und rechte Beschränkung an, um die vertikale Ausrichtung zu ändern, und die obere und untere Beschränkung, um die horizontale Ausrichtung zu ändern. Es werden nur vorhandene Beschränkungen geändert; es werden keine neuen Beschränkungen hinzugefügt.

Ein Beispiel: Komponente A hat eine linke Beschränkung und keine rechte Beschränkung, und Komponente B hat sowohl eine linke als auch eine rechte Beschränkung. Wenn Sie Komponente A und B markieren und dann „Design“ > „Vertikale Mitten ausrichten“ auswählen, passt Flash Builder die linke Beschränkung von Objekt A und die linke und rechte Beschränkung von Objekt B an, um sie auszurichten. Die nicht festgelegte rechte Beschränkung von Objekt A bleibt also weiterhin nicht festgelegt.

Komponenten verschieben

Sie können Komponenten in einem Container, der über absolute Positionierung verfügt, pixelgenau ausrichten, indem Sie die Komponenten mit den Pfeiltasten um jeweils 1 oder 10 Pixel in eine beliebige Richtung verschieben.

Komponenten um 1 Pixel verschieben

- Wählen Sie eine oder mehrere Komponenten im Designmodus des MXML-Editors aus und drücken Sie eine Pfeiltaste.

Komponenten um 10 Pixel verschieben

- Wählen Sie eine oder mehrere Komponenten im Designmodus des MXML-Editors aus und drücken Sie bei gedrückter Umschalttaste eine Pfeiltaste.

Wenn Sie die Pfeiltaste gedrückt halten, wird die Komponente kontinuierlich verschoben.

Containerrahmen ausblenden

Flash Builder zeigt im Designmodus des MXML-Editors standardmäßig Containerrahmen an. Sie können diese Rahmen jedoch auch ausblenden.

- Wählen Sie „Design“ > „Containerrahmen anzeigen“.

  Mit diesem Befehl lassen sich Containerrahmen ein- und ausblenden. Durch erneutes Auswählen werden die Rahmen eingeblendet.

Komponenteneigenschaften festlegen

Sie legen Eigenschaften von Komponenten im Designbereich oder in der Eigenschaftenansicht fest.

Von einer Komponente angezeigten Text bearbeiten

- Um den von einer Komponente angezeigten Text (z. B. ein Label- oder TextInput-Steuerelement) zu bearbeiten, doppelklicken Sie auf die Komponente und geben die Änderungen ein.

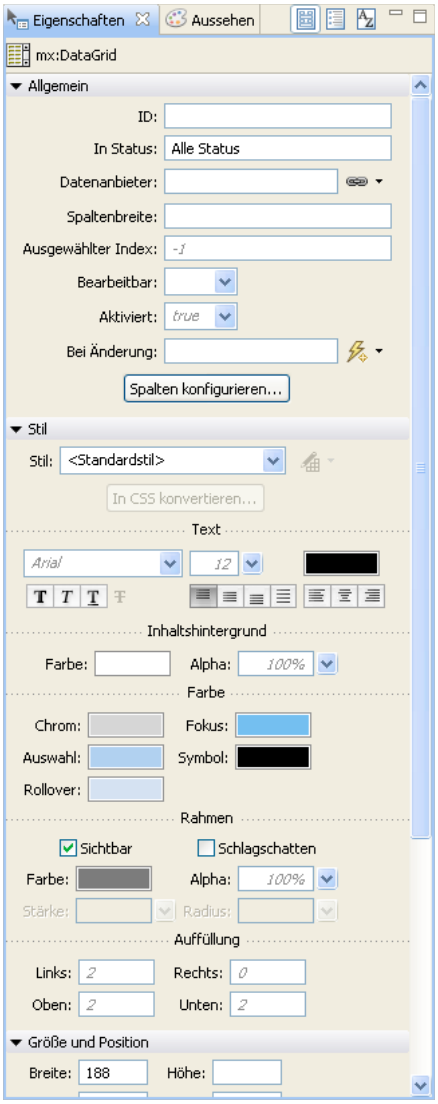
Text im Feld „ID“ ändern

Wenn Sie Text im Feld „ID“ ändern, werden Sie aufgefordert, alle Verweise im Arbeitsbereich mit der neuen ID zu aktualisieren. Sie können dieses Dialogfeld auf der Voreinstellungsseite für den Designmodus unterdrücken:

1. Öffnen Sie das Fenster „Voreinstellungen“.
2. Wählen Sie „Flash Builder“ > „Editoren“ > „Designmodus“.
3. Aktivieren bzw. deaktivieren Sie die Option „Verweise beim Ändern von IDs in der Flex-Eigenschaftenansicht aktualisieren“.

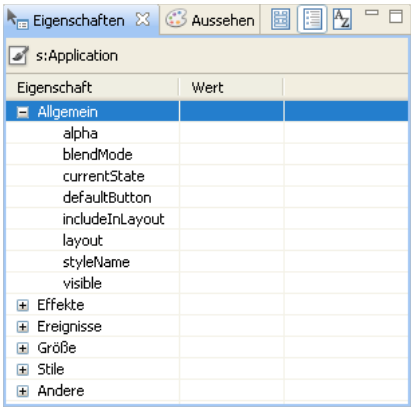
Andere Eigenschaften einer Komponente festlegen

- Wählen Sie die Komponente und definieren Sie ihre Eigenschaften in der Eigenschaftenansicht („Fenster“ > „Ansicht anzeigen“ >„Andere“ > „Flash Builder“ > „Eigenschaften“).



Eigenschaftenansicht einer DataGrid-Komponente

Um die Eigenschaften in der Kategorie- oder alphabetischen Ansicht festzulegen, klicken Sie auf die Ansichtsschaltflächen in der Symbolleiste:



Kategorieansicht einer Button-Komponente

Hinweis: *Um die letzte Änderung zu übernehmen, drücken Sie die Eingabe- oder Tabulatortaste oder klicken außerhalb der Ansicht.*

Komponenten in andere MXML-Dateien kopieren

Sie können Komponenten aus einer MXML-Datei visuell kopieren und in eine andere MXML-Datei einfügen.

1. Öffnen Sie die beiden MXML-Dateien im Designmodus des MXML-Editors, sofern sie noch nicht geöffnet sind.
2. Wählen Sie die Komponente oder Komponenten in einer Datei. Wählen Sie dann „Bearbeiten“ > „Kopieren“.
3. Wechseln Sie zur anderen Datei, klicken Sie in den gewünschten Container und wählen Sie „Bearbeiten“ > „Einfügen“.

Komponenten löschen

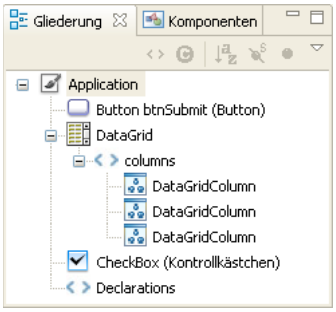
Sie können Komponenten aus der Benutzeroberfläche löschen. Wählen Sie eine Komponente aus, und führen Sie eine der folgenden Aktionen aus:

- Drücken Sie die Entf-Taste auf Ihrer Tastatur.
- Wählen Sie im Kontextmenü einer Komponente den Befehl „Löschen“.
- Wählen Sie im Flash Builder-Menü „Bearbeiten“ den Befehl „Löschen“.

Struktur einer MXML-Datei inspizieren

In der Gliederungsansicht („Fenster“ > „Gliederung“) im Designmodus können Sie die Struktur des Designs inspizieren und eine oder mehrere Komponenten auswählen. Wenn Sie mehrere Ansichtsstatus haben, zeigt die Gliederungsansicht Ihnen die Struktur des aktuellen Ansichtsstatus.

1 Öffnen Sie die gewünschte MXML-Datei im Designmodus (sofern sie noch nicht geöffnet ist) und wählen Sie die Gliederungsansicht.



2 Wählen Sie in der Gliederungsansicht eine oder mehrere Komponenten aus.

Die in der Gliederungsansicht gewählten Komponenten werden auch im Designmodus des Editors gewählt.

Im Quellmodus des Editors wird nur die erste in der Gliederungsansicht gewählte Komponente auch im Editor ausgewählt.

Verwandte Themen

Flash Builder Tips and Tricks (Developer Center-Artikel)

Mit Komponenten im MXML-Quellmodus arbeiten

Im Quellmodus des MXML-Editors unterstützt Sie die Inhaltshilfe beim Hinzufügen von standardmäßigen Flex-Containern und Steuerelementen zur Benutzeroberfläche.

Das folgende Beispiel demonstriert, wie Sie mit der Inhaltshilfe einen `<s:VGroup>`-Container in einen `<s:HGroup>`-Container einfügen. Da HGroup eine Spark-Komponente ist, empfiehlt die Inhaltshilfe einen VGroup-Container und nicht `<mx:VBox>`.

1 Öffnen Sie eine MXML-Datei im Quellmodus des MXML-Editors.

Die MXML-Datei kann die Hauptanwendungsdatei (eine Datei mit einem Anwendungscontainer) oder eine benutzerdefinierte MXML-Kompomentendatei sein.

2 Platzieren Sie den Einfügepunkt in das Tag des übergeordneten Containers.

Um beispielsweise einen VGroup-Container in einen übergeordneten HGroup-Container einzufügen, platzieren Sie den Einfügepunkt hinter das öffnende Tag `<s:HGroup>`:

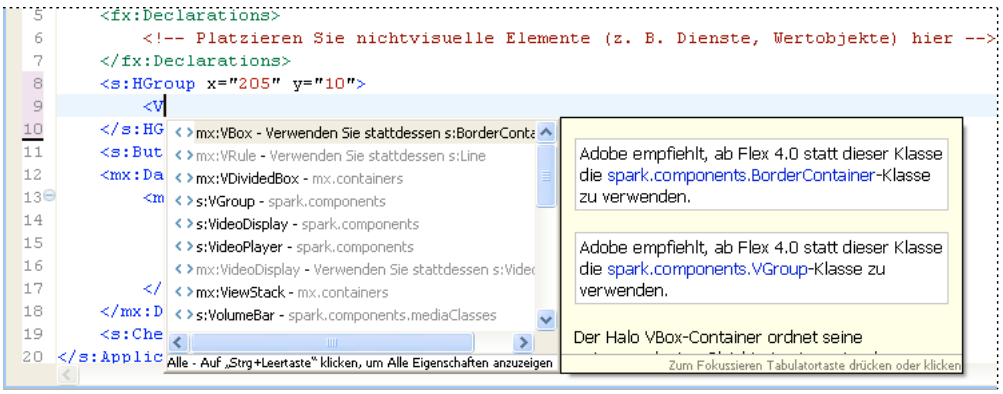
```
<s:HGroup>
    insertion point here
</s:HGroup>
```

3 Geben Sie das Tag für die Komponente ein.

Während Sie das Tag eingeben, wird die Inhaltshilfe mit Vorschlägen für mögliche Einträge angezeigt. Empfohlene Komponenten werden in der normalen schwarzen Schriftfarbe angezeigt. Nicht empfohlene Komponenten werden in grauer Farbe angezeigt.

In diesem Beispiel ist VGroup empfohlen und VBox ist nicht empfohlen.

4 Wählen Sie das Tag gegebenenfalls mithilfe der Pfeiltasten im Menü aus und drücken Sie dann die Eingabetaste.



Neben den Standardkomponenten enthält die Inhaltshilfe die benutzerdefinierten Komponenten, die Sie in separaten MXML- und ActionScript-Dateien definiert und im aktuellen Projekt oder im Quellpfad des aktuellen Projekts gespeichert haben.

Außerdem kann die Inhaltshilfe Eigenschaften, Ereignisse, Effekte und Stile vorschlagen. Drücken Sie Strg+Leertaste, um zwischen den Empfehlungen in der Inhaltshilfe zu wechseln.

Sie können den Typ und die Reihenfolge der Empfehlungen in der Inhaltshilfe ändern. Wählen Sie im Dialogfenster „Voreinstellungen“ die Optionen „Flash Builder“ > „Editoren“ > „MXML-Code“ > „Erweitert“.

Verwandte Themen

„Inhaltshilfe“ auf Seite 49

„Benutzerdefinierte MXML-Komponenten erstellen“ auf Seite 239

Editoren Dateitypen zuordnen

Sie können Editoren verschiedene Dateitypen in der Workbench zuordnen.

1 Wählen Sie „Fenster“ > „Voreinstellungen“.

2 Klicken Sie auf das Pluszeichen, um die Kategorie „Allgemein“ zu erweitern.

3 Klicken Sie auf das Pluszeichen, um die Kategorie „Editoren“ zu erweitern, und wählen Sie dann „Dateizuordnungen“ aus.

4 Wählen Sie aus der Liste „Dateitypen“ einen Dateityp aus.

Um einen Dateityp zur Liste hinzuzufügen, klicken Sie auf „Hinzufügen“, geben den neuen Dateityp im Dialogfeld „Dateityp hinzufügen“ ein und klicken dann auf „OK“.

5 Wählen Sie in der Liste „Zugeordnete Editoren“ den Editor aus, der diesem Dateityp zugeordnet werden soll.

Um einen internen oder externen Editor zur Liste hinzuzufügen, klicken Sie auf „Hinzufügen“ und nehmen die entsprechenden Angaben im Dialogfeld vor.

6 Klicken Sie auf „OK“.

Die standardmäßigen Voreinstellungen des Editors können Sie in einer der Navigationsansichten im Kontextmenü für beliebige Ressourcen überschreiben. Wählen Sie im Kontextmenü „Öffnen mit“.

Dateien außerhalb der Workbench bearbeiten

Sie können eine MXML- oder ActionScript-Datei in einem externen Editor bearbeiten und diese Datei dann in Flash Builder verwenden. Die Workbench führt alle notwendigen Erstellungs- oder Aktualisierungsschritte durch, um die von Ihnen außerhalb der Workbench vorgenommenen Änderungen an der Datei zu verarbeiten.

Außerhalb der Workbench bearbeitete MXML- oder ActionScript-Datei aktualisieren

1 Bearbeiten Sie die MXML- oder ActionScript-Datei im gewünschten externen Editor.

2 Speichern und schließen Sie die Datei.

3 Starten Sie Flash Builder.

4 Wählen Sie in der Workbench in einer der Navigationsansichten im Kontextmenü „Aktualisieren“.

Wenn Sie regelmäßig mit externen Editoren arbeiten, können Sie die automatische Aktualisierungsoption aktivieren. Wählen Sie hierzu „Fenster“ > „Voreinstellungen“ aus, erweitern Sie die Kategorie „Allgemein“, wählen Sie „Arbeitsbereich“ aus und aktivieren Sie die Option „Automatisch aktualisieren“. Wenn Sie diese Option aktivieren, aktualisiert die Workbench automatisch extern an der Datei vorgenommene Änderungen. Wie schnell diese Aktualisierung erfolgt, hängt von der Plattform ab.

Mehrere Editoren nebeneinander anordnen

In der Workbench können Sie mehrere Dateien in mehreren Editoren öffnen. Im Gegensatz zu Ansichten können Editoren jedoch nicht aus der Workbench herausgezogen werden, um neue Fenster zu erstellen. Sie können aber Editoren im Editorbereich nebeneinander anordnen, damit Sie Quelldateien nebeneinander sehen können.

1 Öffnen Sie zwei oder mehr Dateien im Editorbereich.

2 Wählen Sie eine der Editor-Registerkarten aus.

3 Ziehen Sie den Editor über den linken, rechten, oberen oder unteren Rahmen des Editorbereichs.

Der Zeiger ändert sich in einen Ablagecursor, der anzeigt, wo der Editor erscheint, wenn Sie die Maustaste loslassen.

4 (Optional) Ziehen Sie die Rahmen des Editorbereichs der einzelnen Editoren, um ihre Größe wie gewünscht zu ändern.

Ansichten oder Editoren maximieren

Es gibt mehrere Möglichkeiten, eine Ansicht oder einen Editor zu maximieren (wiederherzustellen), sodass das ganze Fenster der Workbench ausgefüllt wird.

- Wählen Sie im Kontextmenü der Ansicht oder in der Titelleiste des Editors die Option „Maximieren“(„Wiederherstellen“).
- Doppelklicken Sie auf die Registerkarte einer Ansicht.
- Wählen Sie im Flash Builder-Menü „Fenster“ > „Editor maximieren/wiederherstellen“.
- Klicken Sie auf die Symbole zum Maximieren/Wiederherstellen in der rechten oberen Ecke der Ansicht bzw. des Editors.

Projekte in Flash Builder

Flash Builder gruppiert die Ressourcen (Ordner und Dateien), die eine Anwendung ausmachen, in einen Container namens *Projekt*.

Verwalten Sie Projekte mit dem Paket-Explorer. Über ihn können Sie Ressourcen hinzufügen, bearbeiten und löschen. Sie können auch Projekte in einem Arbeitsbereich schließen, Ressourcen importieren und Links zu externen Ressourcen erstellen.

Projekttypen

Je nachdem, welchen Anwendungstyp Sie erstellen, unterstützt Flash Builder die folgenden Projekttypen.

Flex-Projekte

Verwenden Sie ein Flex-Projekt, um entweder eine Webanwendung oder eine Desktopanwendung, die auf dem Flex-Framework basieren, zu erstellen. Eine Webanwendung wird in Adobe Flash Player und eine Desktopanwendung in Adobe AIR ausgeführt. Beim Erstellen des Projekts legen Sie fest, ob das Projekt für Web- oder Desktopanwendungen bestimmt ist.

Ein Flex-Projekt enthält eine Gruppe von Eigenschaften, die steuern, wie die Anwendung erstellt wird, wo die erstellte Anwendung gespeichert und wie das Debuggen gehandhabt wird. Die Eigenschaften steuern auch die Beziehungen zu anderen Projekten im Arbeitsbereich.

Ein Flex-Projekt enthält eine standardmäßige MXML-Anwendungsdatei. Andere MXML-Dateien im Projekt können auch Anwendungsdateien sein.

Siehe „Flex-Projekte“ auf Seite 90 und „Entwickeln von Anwendungen mit Flex“ auf Seite 185

Flex-Mobilprojekte

Verwenden Sie ein Flex-Mobilprojekt, um eine für eine Mobilplattform bestimmte AIR-Anwendung zu erstellen. Die Anwendung basiert auf dem Flex-Framework. Verwenden Sie Flash Builder, um Mobilanwendungen vom Desktop aus oder auf einem Gerät in der Vorschau anzuzeigen, zu debuggen oder zu profilieren.

Ein Flex-Mobilprojekt enthält eine einzelne standardmäßige MXML-Anwendungsdatei. Normalerweise beinhaltet eine Mobilanwendung eine Gruppe von Ansichtskomponenten, die Inhalt auf einem Gerät anzeigen. Die standardmäßige MXML-Anwendungsdatei startet die standardmäßige Ansichtskomponente.

Flash Builder verwendet den AIR Debug Launcher (ADL), um Mobilanwendungen auf dem Desktop in der Vorschau anzuzeigen. Obwohl der ADL keine echte Emulation ist, wird darin in diesem Programm das Anwendungslayout und -verhalten angezeigt. Zudem stehen Optionen zum Drehen der Anwendung zur Verfügung.

Sie können eine Mobilanwendung auf einem mit dem USB-Port des Entwicklungscomputers verbundenen Gerät in der Vorschau anzeigen. Für die Vorschau auf einem Gerät exportiert Flash Builder die Anwendung auf das Gerät.

Siehe „Flex-Mobilprojekte“ auf Seite 92 und Entwickeln von Mobilanwendungen mit Flex und Flash Builder.

ActionScript-Projekte

Verwenden Sie ein ActionScript-Projekt, um Web- oder Desktopanwendungen, die entweder auf Flash- oder Adobe AIR-APIs basieren, zu erstellen. Beim Erstellen des Projekts legen Sie fest, ob das Projekt für eine Web- oder Desktopanwendung bestimmt ist.

Da diese Projekte kein MXML zur Definition einer Benutzeroberfläche verwenden, können Sie das Layout und Design der Anwendung nicht im Designmodus anzeigen. Sie arbeiten exklusiv im Quelleditor und setzen die Debugwerkzeuge nach Bedarf ein. Anschließend erstellen Sie SWF-Anwendungsdateien aus dem Projekt, um die Anwendung anzuzeigen und zu testen.

Wenn Sie ein ActionScript-Projekt oder eine eigenständige ActionScript-Datei mit Funktionen, einer Klasse oder einer Schnittstelle erstellen, wird in der Flex-Entwicklungsperspektive auch der ActionScript-Editor angezeigt. Die primären Hilfsansichten des ActionScript-Editors sind die Gliederungs- und die Problemansicht.

Siehe „ActionScript-Projekte“ auf Seite 93.

ActionScript-Mobilprojekte

Verwenden Sie ein ActionScript-Mobilprojekt, um auf der Adobe AIR-API basierende Mobilanwendungen zu erstellen. Beim Erstellen eines Projekts geben Sie eine Zielmobilplattform und Mobilanwendungseinstellungen an. Verwenden Sie Flash Builder, um Mobilanwendungen vom Desktop aus oder auf einem Gerät in der Vorschau anzuzeigen.

Flash Builder verwendet den AIR Debug Launcher (ADL), um Mobilanwendungen auf dem Desktop in der Vorschau anzuzeigen, zu debuggen oder zu profilieren. Obwohl der ADL keine echte Emulation ist, wird darin in diesem Programm das Anwendungslayout und -verhalten angezeigt. Zudem stehen Optionen zum Drehen der Anwendung zur Verfügung.

Sie können eine Mobilanwendung auf einem mit dem USB-Port des Entwicklungscomputers verbundenen Gerät in der Vorschau anzeigen. Für die Vorschau auf einem Gerät exportiert Flash Builder die Anwendung auf das Gerät. Verwenden Sie Flash Builder, um die auf ein Gerät exportierte Anwendung zu debuggen.

Siehe „ActionScript-Mobilprojekte erstellen“ auf Seite 93.

Flex-Bibliotheksprojekte

Mit einem Flex-Bibliotheksprojekt können Sie benutzerdefinierte Codebibliotheken erstellen, die Sie in anderen Anwendungen nutzen bzw. an andere Entwickler weitergeben können. Bibliotheksprojekte werden normalerweise zum Verpacken und Verteilen von Komponenten und anderer Ressourcen für andere Entwickler verwendet.

Ein Bibliotheksprojekt generiert eine SWC-Datei, die eine Archivdatei für Flex-Komponenten und andere Ressourcen ist.

Siehe „Flex-Bibliotheksprojekte“ auf Seite 234.

Mit Flash Catalyst kompatible Projekte

Benutzen Sie ein mit Flash Catalyst kompatibles Projekt, wenn Sie Projektdateien mit Adobe Flash Catalyst-Projekten verwenden möchten. Mithilfe dieser Funktion können Designer und Entwickler an demselben Projekt zusammenarbeiten.

Siehe „Mit Flash Catalyst kompatible Projekte“ auf Seite 94.

Flash Professional-Projekte

Mit einem Flash Professional-Projekt können Sie in Adobe Flash Professional erstellte FLA- und XFL-Dateien bearbeiten, erzeugen und debuggen. Damit können Flash Professional-Entwickler die in Flash Builder verfügbare Bearbeitungs- und Debug-Umgebung nutzen. Flash Professional-Projekte sind nur in Flash Builder verfügbar, wenn Flash Professional installiert ist.

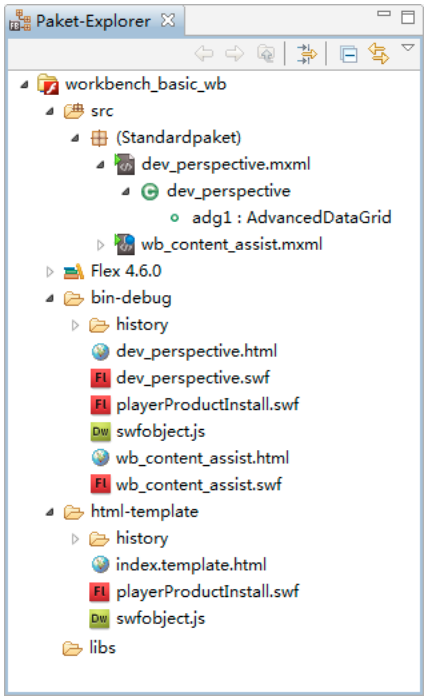
Siehe „Verwenden von Flash Builder mit Flash Professional“ auf Seite 251.

Projekte verwalten

Mit dem Paket-Explorer können Sie Projekten Ressourcen hinzufügen, Ressourcen in Projekte importieren, Projekte exportieren und Ressourcen verschieben oder löschen.

Projekte im Paket-Explorer

Alle Projekte in einem Arbeitsbereich werden im Paket-Explorer angezeigt, wie das folgende Beispiel zeigt. Der Paket-Explorer zeigt die Projekte in einer Strukturansicht entweder hierarchisch oder logisch (flach) geordnet an. Sie verwalten Projekte in dieser Ansicht, indem Sie Ressourcen (Ordner und Dateien) hinzufügen und löschen, externe Ressourcen importieren und verlinken und Ressourcen in andere Projekte im Arbeitsbereich verschieben.



Die Highlights des Paket-Explorer:

- Anzeigen von ActionScript-Paketen in hierarchischer oder flacher Form.

  Festlegen der Paketpräsentation über das Menü des Paket-Explorers.

- Projektbibliotheken werden in zwei Knoten auf oberster Ebene dargestellt. Ein Knoten repräsentiert das Flex SDK, der zweite Knoten referenzierte Bibliotheken.

  Sie können den Inhalt einer Bibliothek erweitern und Editoren öffnen, um Anlagen anzuzeigen.

- Fehler und Warnungen auf Knoten im Paket-Explorer weisen auf Probleme in einem Paket hin.
- Sie können die Sichtbarkeit von Projekten und Ressourcen beschränken.

  Sie können ein Arbeitsset (eine Sammlung von Ressourcen) und Anzeigefilter erstellen und Ressourcen nach Namen und Typen sortieren. Diese Optionen stehen über das Menü des Paket-Explorers zur Verfügung. Weitere Informationen zum Ändern von Ansichten finden Sie unter „Workbench anpassen“ auf Seite 266.

- Sie können ActionScript-, MXML- und CSS-Dateien erweitern und eine Baumansicht des Inhalts der jeweiligen Datei sehen.

Sie können die Projektressourcen zur Bearbeitung vom Paket-Explorer aus öffnen. Beispielsweise können Sie MXML und ActionScript in `<fx:Script>` -Blöcken und CSS in `<fx:Style>`-Blöcken bearbeiten. Oder Sie schalten in den Designmodus um und nutzen die visuellen Bearbeitungsmöglichkeiten der Komponenten und Steuerelemente, um das Layout und Verhalten von Anwendungen zu erstellen. Weitere Informationen zum Arbeiten mit den Flash Builder-Editoren finden Sie unter „Editoren in Flash Builder“ auf Seite 21 und „Benutzeroberflächen erstellen“ auf Seite 186.

Anschließend fügen Sie Projekte, Dateien und Ordner hinzu und organisieren und verwalten sie bei Bedarf. Weitere Informationen finden Sie unter „Projekte in Flash Builder“ auf Seite 35.

Die meisten Menübefehle, die Sie im Paket-Explorer verwenden, sind auch über das Kontextmenü des Explorers aufrufbar.

Verwandte Themen

„Projekte verwalten“ auf Seite 37

„Projektressourcen“ auf Seite 42

Projekte zwischen Arbeitsbereichen verschieben

Um ein Projekt aus einem Arbeitsbereich in einen anderen zu verschieben, verwenden Sie eine Kombination aus Lösch- und Importvorgängen. Wenn Sie ein Projekt aus einem Arbeitsbereich löschen, können Sie es aus dem Arbeitsbereich entfernen, aber im Dateisystem belassen (siehe „Projekt löschen“ auf Seite 39). Nachdem Sie ein Projekt aus einem Arbeitsbereich entfernt haben, können Sie es in einen anderen Arbeitsbereich importieren.

SDK für ein Projekt festlegen

Wenn Sie ein Flex-Projekt anlegen, können Sie angeben, welches Flex SDK verwendet werden soll. Sie können die SDK-Einstellungen aber zu einem späteren Zeitpunkt über „Projekt“ > „Eigenschaften“ > „Flex Compiler“ > „Bestimmtes SDK verwenden“ ändern.

Wenn Sie das Projekt mit einer Version des Flex SDKs kompilieren möchten, die in Ihrer Flash Builder-Installation nicht verfügbar ist, können Sie das SDK herunterladen und zu Ihrer Installation hinzufügen. Laden Sie beispielsweise das SDK herunter und fügen Sie es mithilfe von „Projekt“ > „Eigenschaften“ > „Flex Compiler“ > „Flex SDKs konfigurieren“ zu Flash Builder hinzu.

Projekt löschen

Wenn Sie ein Projekt löschen, entfernen Sie das Projekt aus dem aktuellen Arbeitsbereich. Sie können das Projekt auch gleichzeitig aus dem Dateisystem entfernen.

Statt das Projekt aus dem Arbeitsbereich zu löschen, können Sie das Projekt schließen. Wenn Sie das Projekt schließen, behalten Sie dadurch einen Verweis auf das Projekt im Arbeitsbereich und geben außerdem einige Systemressourcen frei. Weitere Informationen hierzu finden Sie unter „Projekte schließen und öffnen“ auf Seite 39.

1 Wählen Sie im Paket-Explorer das Projekt aus, das gelöscht werden soll.

2 Wählen Sie im Hauptmenü „Bearbeiten“ > „Löschen“.

3 Wählen Sie eine Option aus:

Inhalt unter Ordner ebenfalls löschen Das Projekt wird komplett aus dem Arbeitsbereich und dem Dateisystem entfernt.

Inhalt nicht löschen Das Projekt wird aus dem Arbeitsbereich, aber nicht aus dem Dateisystem entfernt.

Projekte schließen und öffnen

Um Speicherplatz zu sparen und die zur Erstellung von Projekten benötigte Zeit zu verkürzen, ohne ein Projekt zu löschen, können Sie das Projekt schließen. Wenn Sie ein Projekt schließen, klappen Sie das Projekt und seine Ressourcen ein; der Name bleibt aber weiterhin im Paket-Explorer sichtbar. Ein geschlossenes Projekt erfordert weniger Speicherplatz als ein geöffnetes Projekt und wird aus Erstellungsprozessen ausgeschlossen. Sie können das geschlossene Projekt leicht wieder öffnen.

1 Wählen Sie im Flex Paket-Explorer das Projekt aus, das geschlossen oder erneut geöffnet werden soll.

2 Wählen Sie im Kontextmenü des Paket-Explorers „Projekt schließen“ bzw. „Projekt öffnen“.

Hauptanwendungsdatei ändern

Wenn Sie ein Projekt anlegen, wird für Sie die Hauptanwendungsdatei generiert. Diese Datei wird standardmäßig nach dem Projekt benannt. Die Hauptanwendungsdatei ist der Einstiegspunkt in Anwendungen und wird die Basis der SWF-Anwendungsdatei. Wenn Sie der Anwendung Dateien hinzufügen, möchten Sie aber unter Umständen eine andere Datei als Hauptanwendungsdatei auswählen.

Wenn Sie mehrere Dateien als Anwendungsdateien festlegen möchten, sodass jede Anwendungsdatei in einer separaten SWF-Datei erstellt wird, lesen Sie bitte „Projektanwendungsdateien verwalten“ auf Seite 39.

1 Wählen Sie im Paket-Explorer die MXML-Anwendungsdatei aus, die Sie zur Hauptanwendungsdatei machen möchten.

2 Wählen Sie im Kontextmenü des Paket-Explorers „Als Standardanwendung festlegen“.

Sie können die Anwendungsdateien im Projekt verwalten, indem Sie „Projekt“ > „Eigenschaften“ > „Flex-Anwendungen“ (oder ActionScript-Anwendungen, wenn Sie mit einem ActionScript-Projekt arbeiten) auswählen.

Projektanwendungsdateien verwalten

Ein Projekt hat in der Regel eine Hauptanwendungsdatei, die als Einstiegspunkt in die Anwendung dient. Der Flash Builder-Compiler verwendet diese Datei, um die SWF-Anwendungsdatei zu generieren.

Beispiel: Eine komplexe Flex-Anwendung mit vielen benutzerdefinierten MXML-Komponenten, die unterschiedliche, aber zusammengehörige Anwendungselemente darstellen. Sie können eine Anwendungsdatei erstellen, die eine benutzerdefinierte Komponente enthält, und diese Datei dann separat erstellen, ausführen und testen.

Standardmäßig können Sie die Anwendung ausführen, wenn Sie eine MXML-Anwendungsdatei zum Flex-Projekt hinzufügen. Die Anwendungsdatei wird dann zur Liste der Projektanwendungsdateien hinzugefügt. Alle als Anwendungsdateien definierten Dateien müssen sich im Quellordner des Projekts befinden.

Sie können die Anwendungsdateiliste verwalten, indem Sie ein Projekt auswählen und seine Eigenschaften anzeigen.

1 Wählen Sie im Paket-Explorer ein Projekt aus.

2 Wählen Sie im Hauptmenü „Projekt“ > „Eigenschaften“ oder im Kontextmenü „Eigenschaften“.

3 Wählen Sie im Dialogfeld „Projekteigenschaften“ die Option „Flex-Anwendungen“ (oder „ActionScript-Anwendungen“, wenn Sie mit einem ActionScript-Projekt arbeiten).

4 Fügen Sie die gewünschten Anwendungsdateien hinzu bzw. entfernen Sie Anwendungsdateien. Klicken Sie auf „OK“.

Arbeitsbereich ändern

Sie können jeweils nur in einem Arbeitsbereich arbeiten. Wenn Sie Flash Builder installieren und zum ersten Mal starten, werden Sie aufgefordert, einen Arbeitsbereich zu erstellen. Dieser Arbeitsbereich wird der Standardarbeitsbereich. Sie können weitere Arbeitsbereiche erstellen und zwischen ihnen wechseln, indem Sie den gewünschten Arbeitsbereich entweder beim Starten von Flash Builder oder über „Datei“ > „Arbeitsbereich wechseln“ auswählen.

Flex-Projekteigenschaften festlegen

Jedes Flex-Projekt hat eine eigene Gruppe von Eigenschaften. Um diese Eigenschaften festzulegen, wählen Sie das Projekt im Paket-Explorer aus. Wählen Sie dann im Hauptmenü „Projekt“ >„Eigenschaften“. Sie können auch im Kontextmenü des Projekts „Eigenschaften“ wählen.

In Flash Builder können Sie die folgenden projektspezifischen Voreinstellungen festlegen:

Ressource Zeigt allgemeine Informationen zum Projekt, Einstellungen für Textkodierung und den Zeilenbegrenzer des Betriebssystems an.

Erstellungsprogramme Legt fest, welches Erstellungsprogramm verwendet werden soll. Flash Builder enthält ein Standarderstellungsprogramm. Sie können Apache Ant (ein Open-Source-Erstellungsprogramm) verwenden, um Erstellungsskripten zu erstellen oder vorhandene Ant-Erstellungsskripten zu importieren.

Siehe „Builds mit Apache Ant anpassen“ auf Seite 113.

Datenmodell Nur mit ADEP Data Services (früher LiveCycle Data Services genannt) und höher verfügbar. Gibt den Speicherort der Datenmodelldatei an, die Dienst- und Datentypinformationen zu ADEP Data Services enthält.

Daten/Dienste Gibt für Projekte, die auf Datendienste zugreifen an, ob der Standard-Codegenerator für den Zugriff auf Dienste verwendet werden soll. Sie können auch angeben, ob beim Zugreifen auf Dienste eine einzelne Serverinstanz verwendet werden soll.

Weitere Informationen zum Erweitern von Flash Builder auf die Verwendung von benutzerdefinierter Codegenerierung finden Sie unter Adobe Flash Builder erweitern.

Weitere Informationen zum Verwenden einer einzelnen Serverinstanz beim Zugreifen auf Dienste finden Sie unter Verwenden einer einzelnen Serverinstanz.

Flex-Anwendungen Zeigt die Namen der Projektdateien an, die als Anwendungsdateien festgelegt sind und kompiliert, debuggt und als separate Anwendungen ausgeführt werden können.

Siehe „Projektanwendungsdateien verwalten“ auf Seite 39.

Flex-Erstellungspfad Legt den Erstellungspfad fest, der angibt, wo sich externe Quell- und Bibliotheksdateien befinden. Sie können den Erstellungspfad ändern und auch einen anderen Namen für den Ausgabeordner angeben.

Siehe „Projektausgabeordner einrichten“ auf Seite 109 und „Projekte manuell erstellen“ auf Seite 111.

Flex Compiler Legt optionale Compilervoreinstellungen fest, z. B. Generierung einer barrierefreien SWF-Datei, Aktivierung von Compilerwarnungen und Typprüfungen, Angabe zusätzlicher Compilerargumente, Flex SDK-Version und Festlegung von Einstellungen für HTML-Wrapper.

Siehe „Erweiterte Erstellungsoptionen“ auf Seite 111.

Flex Module Legt fest, welche Module für das Projekt erstellt und optimiert werden sollen. Weitere Information zur Verwendung von Modulen in Flash Builder finden Sie unter „Separates Projekt für Module in Flash Builder erstellen“ auf Seite 243.

Flex Server Gibt den Anwendungsservertyp für das Projekt an. Beim Erstellen eines Projekts geben Sie den Anwendungsservertyp an. Hier können Sie den Anwendungsservertyp für ein Projekt ändern. Wenn Sie den Anwendungsservertyp für ein Projekt ändern, können Sie möglicherweise nicht auf zuvor konfigurierte Datendienste zugreifen.

Siehe „Flex-Projekte“ auf Seite 90 und Erstellen eines Flex-Projekts für den Zugriff auf Datendienste.

Flex Design Legt das Design fest, das für alle Anwendungen im Projekt verwendet werden soll. Sie können eines der in Flash Builder verfügbaren Designs angeben oder ein Design importieren.

Siehe „Designs übernehmen“ auf Seite 191.

Projektverweise Listet die Projekte auf, auf die das aktuelle Projekt verweist.

Ausführungs-/Debugeinstellungen Verwaltet die Startkonfigurationseinstellungen.

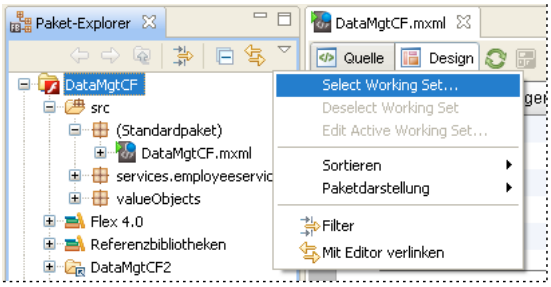
Siehe „Startkonfigurationen verwalten“ auf Seite 123.

Arbeitssets erstellen

Wenn der Arbeitsbereich viele Projekte enthält, können Sie ein Arbeitsset erstellen, um ausgewählte Projekte zusammenzufassen. Sie können dann einzelne Arbeitssets im Paket-Explorer und in der Taskansicht anzeigen und auch in Arbeitssets anstatt im kompletten Arbeitsbereich suchen.

Arbeitsset erstellen

1 Öffnen Sie im Paket-Explorer das Symbolleistenmenü und wählen Sie „Arbeitsset auswählen“.



2 Wählen Sie „Neu“.

Flash Builder bietet zwei Settypen: Haltepunkt (beim Debuggen verwendet) und Ressource.

3 Wählen Sie den Typ „Ressource“ aus und klicken Sie auf „Weiter“.

4 Geben Sie einen Namen für das Arbeitsset ein und wählen Sie dann die Projekte im Arbeitsbereich, die in das Arbeitsset aufgenommen werden sollen.

5 Klicken Sie auf „Fertig stellen“.

Das Arbeitsset wird sofort auf den Paket-Explorer angewendet und es werden nur die Projekte und Ressourcen angezeigt, die im Set enthalten sind.

Alle Projekte im Arbeitsbereich anzeigen

❖ Öffnen Sie im Paket-Explorer das Symbolleistenmenü und wählen Sie „Arbeitssetauswahl aufheben“.

Projektressourcen

Projekte bestehen aus Ressourcen (Ordnern und Dateien), die Sie vom Paket-Explorer aus verwalten können. Projekte befinden sich in einem Arbeitsbereich. Der Paket-Explorer bietet eine logische Repräsentation des Arbeitsbereichs im Dateisystem. Der Paket-Explorer wird jedes Mal aktualisiert, wenn Sie eine Ressource hinzufügen, löschen oder ändern.

Sie können Projektressourcen direkt im Dateisystem bearbeiten und Flash Builder sowie den Paket-Explorer hierbei umgehen.

Dateien und Ordner in einem Projekt erstellen

Eine Anwendung in Adobe® Flex® besteht in der Regel aus einer MXML-Anwendungsdatei, einer oder mehreren Ansichten (nur Mobilprojekte) sowie einer oder mehreren Flex-Standardkomponenten. Darüber hinaus können Sie eine oder mehrere benutzerdefinierte Komponenten in separaten MXML-, ActionScript- oder Flash-Komponenten (SWC)-Dateien definieren. Wenn Sie die Anwendung in überschaubare Abschnitte aufteilen, können Sie jede Komponente unabhängig von den anderen schreiben und testen. Sie können außerdem eine Komponente in der gleichen oder in anderen Anwendungen erneut verwenden, wodurch die Effizienz erhöht wird.

Mit Adobe Flash® Builder™ können Sie benutzerdefinierte MXML- und ActionScript-Komponenten erstellen und sie anschließend in Ihre Anwendungen einfügen. Informationen zum Erstellen von ActionScript-Komponenten finden Sie unter „ActionScript-Klasse erstellen“ auf Seite 43.

Sie können eine MXML-Komponente auch direkt über Code erstellen. Weitere Informationen hierzu finden Sie unter Einfache MXML-Komponenten.

Datei erstellen

1 Wählen Sie im Paket-Explorer „Datei“ > „Neu“ > „Datei“.

2 Wenn Sie mehrere Projekte im Arbeitsbereich haben, wählen Sie das Projekt aus, dem die Datei hinzugefügt werden soll.

3 Geben Sie den Dateinamen ein und klicken Sie auf „Fertig stellen“.

Sie können auch Ordner und Dateien hinzufügen, die sich außerhalb des aktuellen Projekts befinden. Weitere Informationen hierzu finden Sie unter „Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken“ auf Seite 46.

MXML-Anwendungsdateien erstellen

1 Wählen Sie im Paket-Explorer ein Projekt für die MXML-Anwendungsdatei aus.

2 Wählen Sie im Kontextmenü des Paket-Explorers „Neu“ > „MXML-Anwendung“.

3 Die Anwendungsdatei wird standardmäßig im Ordner „src“ erstellt. Sie im Projekt aber auch einen anderen Ordner wählen.

4 Geben Sie einen Namen für die Anwendungsdatei an und wählen Sie ggf. ein Spark-Layout. Weitere Informationen hierzu finden sie unter Spark-Layouts

5 Folgendes gilt nur für Mobilprojekte:

   a Legen Sie eine Anwendungs-ID fest.

   b Legen Sie die Anwendungsvorlage fest.

   Siehe Anwendungsvorlage auswählen

   c Legen Sie die Mobilanwendungsberechtigungen für die Google Android-Plattform fest.

   Siehe Mobilanwendungsberechtigungen auswählen.

   d Legen Sie die Plattformeinstellungen fest.

   Siehe Plattformeinstellungen auswählen.

   e Legen Sie die Anwendungseinstellungen fest.

   Siehe Anwendungseinstellungen auswählen.

6 Klicken Sie auf „Fertig stellen“.

ActionScript-Klasse erstellen

In Flash Builder können Sie einen Assistenten verwenden, um schnell ActionScript-Klassen für Flex- und ActionScript-Projekte zu erstellen. Mit dem Assistenten lassen sich auch leicht Codeabschnitte für Funktionen generieren, die implementiert werden müssen.

1 Wählen Sie „Datei“ > „Neu“ > „ActionScript-Klasse“.

2 Geben Sie die Grundeigenschaften der neuen Klasse im Dialogfeld an und klicken Sie dann auf „Fertig stellen“.

   Wenn Sie auf „Fertig stellen“ klicken, speichert Flash Builder die Datei im angegebenen Paket und öffnet sie im Codeeditor.

   Wenn Sie die Datei im aktuellen Projekt oder im Quellpfad des aktuellen Projekts gespeichert haben, zeigt Flash Builder auch die Komponente in der Komponentenansicht an, sodass Sie diese schnell in Anwendungen einfügen können. Weitere Informationen hierzu finden Sie unter „Komponenten im MXML-Designmodus hinzufügen“ auf Seite 24.

3 Schreiben Sie die Definition der ActionScript-Klasse.

   Weitere Informationen finden Sie unter Einfache visuelle Komponenten in ActionScript erstellen.

ActionScript-Schnittstelle erstellen

In Flash Builder können Sie ActionScript-Schnittstellen für Flex- und ActionScript-Projekte mithilfe eines Assistenten schnell erstellen. Eine *Schnittstelle* ist eine Sammlung von Konstanten und Methoden, die verschiedene Klassen gemeinsam nutzen können.

1 Wählen Sie „Datei“ > „Neu“ > „ActionScript-Schnittstelle“.

2 Geben Sie die Grundeigenschaften der neuen Schnittstelle im Dialogfeld an und klicken Sie dann auf „Fertig stellen“.

3 Fügen Sie Konstanten oder Methoden zur ActionScript-Schnittstelle hinzu, die von verschiedenen Klassen genutzt wird.

Skinfähige ActionScript-Komponente erstellen

In Flash Builder können Sie mit einem Assistenten schnell ActionScript-Komponenten für Flex-Projekte erstellen.

1 Wählen Sie „Datei“ > „Neu“ > „Skinfähige ActionScript-Komponente“.

2 Geben Sie die Grundeigenschaften der neuen skinfähigen Komponente an und klicken Sie dann auf „Fertig stellen“.

Flash Builder erstellt eine ActionScript-Klasse, die von der Basiskomponente oder der skinfähigen Komponente ausgeht. Standardmäßig ist die Basiskomponente `spark.components.supportClasses.SkinnableComponent`. Sie können die Basiskomponente jedoch ändern, indem Sie eine andere Komponente verwenden, die eine Unterklasse der skinfähigen Komponente darstellt.

Außerdem können Sie ActionScript-Schnittstellen hinzufügen, die von der neuen skinfähigen Komponente implementiert werden können. Weitere Informationen zum Erstellen von ActionScript-Schnittstellen finden Sie unter „ActionScript-Schnittstelle erstellen“ auf Seite 43.

Die `SkinnableComponent` wird am gewählten Speicherort und im gewählten Paket erstellt. In die MXML-Hauptdatei wird automatisch ein Verweis auf die Komponente eingefügt.

Ordner in einem Projekt erstellen

Sie können jederzeit neue Ordner zum Projekt hinzufügen. Beispielsweise können Sie einen Ordner erstellen, um dort alle Datenmodelle zu speichern. Oder Sie können alle Elemente organisieren, die das visuelle Design Ihrer Anwendung ausmachen.

1 Wählen Sie im Paket-Explorer „Datei“ > „Neu“ > „Ordner“ aus.

2 Wenn Sie mehrere Projekte im Arbeitsbereich haben, wählen Sie das Projekt aus, dem der eigenständige Ordner hinzugefügt werden soll.

Wenn Sie den Ordner im Quellpfadordner erstellen, wird er wie ein Paketname behandelt. Sie können dann in diesen Ordner Quelldateien ablegen, die vom Compiler erkannt werden.

Wenn Sie den Ordner außerhalb des Quellpfadordners erstellen, können Sie ihn später zum Stamm einer Paketstruktur machen, indem Sie ihn dem Quellpfad hinzufügen. Nachdem Sie diese Schritte durchgeführt haben, wählen Sie „Projekt“ > „Eigenschaften“ und dann „Flex-Erstellungspfad“. Klicken Sie auf „Ordner hinzufügen“ und navigieren Sie zum neu erstellten Ordner.

3 Geben Sie den Ordnernamen ein und klicken Sie auf „Fertig stellen“.

Ordner und Dateien löschen

Wenn Sie Ordner und Dateien aus dem Projekt löschen, werden sie aus dem Arbeitsbereich und damit aus dem Dateisystem entfernt.

Hinweis: *Wenn Sie eine verlinkte Ressource löschen, löschen Sie nur den Link aus dem Projekt, nicht die Ressource selbst (siehe „Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken“ auf Seite 46). Wenn Sie aber einen Link zu einem Ordner angelegt haben und Dateien in diesem Ordner löschen, werden die Dateien aus dem Dateisystem entfernt.*

1 Wählen Sie im Paket-Explorer die Ressource aus, die gelöscht werden soll.

2 Wählen Sie „Bearbeiten“ > „Löschen“ aus oder drücken Sie die Entf-Taste und klicken Sie auf „Ja“.

Die Ressource wird aus dem Dateisystem gelöscht.

In der Workbench nach Ressourcen suchen

Flash Builder enthält eine Suchfunktion, mit der Sie Ressourcen schnell finden können.

So durchsuchen Sie das aktuelle Dokument:

1 Öffnen Sie das Dokument, das durchsucht werden soll.

2 Führen Sie einen der folgenden Schritte aus:

- Drücken Sie Strg+F (Windows) bzw. Befehlstaste+F (Mac OS).
- Wählen Sie „Bearbeiten“ > „Suchen/Ersetzen“.

3 Geben Sie den Textstring ein, der gesucht werden soll.

4 (Optional) Geben Sie den Textstring ein, der den gesuchten Text ersetzen soll.

5 (Optional) Legen Sie erweiterte Suchkriterien fest.

6 Klicken Sie auf „Suchen“, „Ersetzen“, „Alle ersetzen“ oder „Ersetzen/Suchen“.

Wenn der Textstring im Dokument gefunden wird, wird er hervorgehoben und (optional) ersetzt.

Hinweis: *Um die nächste Fundstelle zu suchen, drücken Sie Strg+J (Windows) bzw. Befehlstaste+J (Mac OS).*

Um alle Ressourcen im Arbeitsbereich zu durchsuchen, setzt Flash Builder erweiterte Suchfunktionen ein, die über das übliche Suchen und Ersetzen hinausgehen. Mit Flash Builder können Sie Verweise oder Erklärungen zu Kennungen in ActionScript- und MXML-Dateien, Projekten oder Arbeitsbereichen suchen und markieren. Weitere Informationen hierzu finden Sie unter „Verweise suchen und Code refaktorieren“ auf Seite 77.

Dateien suchen

❖ Über „Suchen“ > „Suchen“ können Sie komplexe Dateisuchen durchführen.

Hinweis: *Klicken Sie auf „Anpassen“, um zu definieren, welche Suchregister im Dialogfeld „Suchen“ verfügbar sein sollen.*

Suchansicht

Die Suchansicht zeigt die Ergebnisse der Suche an.

Datei aus der Liste öffnen

❖ Doppelklicken Sie auf die Datei.

Datei aus der Liste entfernen

❖ Wählen Sie die Datei aus, die entfernt werden soll, und klicken Sie auf „Ausgewählte Übereinstimmungen entfernen“.

Alle Dateien aus der Liste entfernen

❖ Klicken Sie auf „Alle Übereinstimmungen entfernen“.

Zwischen übereinstimmenden Dateien navigieren

❖ Klicken Sie auf „Nächste Übereinstimmung anzeigen“ bzw. „Vorherige Übereinstimmung anzeigen“.

Frühere Suchvorgänge anzeigen

❖ Klicken Sie auf den Abwärtspfeil neben „Vorherige Suchvorgänge anzeigen“ und wählen Sie einen Suchvorgang aus der Pulldownliste aus.

Zur Suchansicht zurückkehren, nachdem sie geschlossen wurde

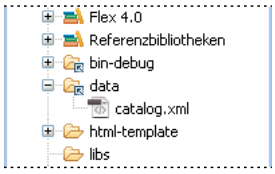
1 Wählen Sie „Fenster“ > „Ansicht anzeigen“ > „Andere“.

2 Erweitern Sie die Kategorie „Allgemein“, wählen Sie „Suchen“ aus und klicken Sie auf „OK“.

Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken

Sie können Ressourcen verlinken, die sich außerhalb des Projekts und des Arbeitsbereichs befinden. Ordner und Dateien können von einem beliebigen Ort im Dateisystem verlinkt werden. Diese Option ist praktisch, wenn Sie Ressourcen haben, die in mehreren Projekten verwendet werden. Beispielsweise können Sie eine aus benutzerdefinierten Flex-Komponenten oder ActionScript-Dateien bestehende Bibliothek in vielen verschiedenen Flex-Projekten verwenden.

Ordner, die verlinkte Ressourcen enthalten, sind im Paket-Explorer markiert (wie im folgenden Beispiel gezeigt), damit Sie zwischen normalen und verlinkten Ressourcen unterscheiden können.



Andere Beispiele für verlinkte Ressourcen sind Ordner mit Grafikdateien oder Situationen, in denen sich der Ausgabeordner nicht im Stammordner des Projekts befindet.

Wenn Sie mit gemeinsam genutzten Ressourcen arbeiten, wirken sich die von Ihnen vorgenommenen Änderungen an Quellordnern und Dateien auf alle Projekte aus, mit denen diese Ordner und/oder Dateien verlinkt sind. Seien Sie vorsichtig, wenn Sie verlinkte Ressourcen aus Projekten löschen. In einigen Fällen löschen Sie nur den Linkverweis und in anderen löschen Sie die eigentliche Quelle. Weitere Informationen hierzu finden Sie unter „Ordner und Dateien löschen“ auf Seite 44.

Hinweis: *Es wird empfohlen, andere Projekte mit dem Bibliotheksprojekt zu verlinken. Verlinkte Ressourcen werden nur bei Bibliotheken von Drittanbietern mit einer SWC-Datei verwendet.*

Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken

1. Wählen Sie im Paket-Explorer das Projekt aus, dem verlinkte Ressourcen hinzugefügt werden sollen.
2. Wählen Sie „Datei“ > „Neu“ > „Ordner“ (oder „Datei“).
3. Wählen Sie das Projekt oder den Projektordner aus, dem die verlinkten Ressourcen hinzugefügt werden sollen.
4. Geben Sie den Ordner- oder Dateinamen ein. Der Ordner- oder Dateiname, den Sie eingeben, kann anders als der Name des Ordners oder der Datei sein, den bzw. die Sie verlinken.
5. Klicken Sie auf die Schaltfläche „Erweitert“.
6. Aktivieren Sie die Option „Link zu Datei im Dateisystem“. Geben Sie den Pfad der Ressource ein oder klicken Sie auf „Durchsuchen“ und wählen Sie den Pfad aus.
7. Klicken Sie auf „Fertig stellen“, um die Ressource mit dem Projekt zu verlinken.

Pfadvariablen zum Verlinken von Ressourcen verwenden

Anstatt Ressourcen durch Eingabe des vollständigen Pfads zum lokalen Ordner oder Netzlaufwerk, auf dem Sie die Dateien speichern, zu verlinken, können Sie Pfadvariablen definieren. Weitere Informationen hierzu finden Sie unter „Pfadvariablen erstellen“ auf Seite 111.

1. Wählen Sie im Paket-Explorer das Projekt aus, dem verlinkte Ressourcen hinzugefügt werden sollen.

   Pfadvariablen können auch in bestimmten Projekteinstellungen wie z. B. im Bibliothekspfad und im Quellpfad verwendet werden.

2. Wählen Sie „Datei“ > „Neu“ > „Ordner“ (oder „Datei“, wenn Sie Dateien hinzufügen möchten).

3 Wählen Sie das Projekt oder den Projektordner aus, dem die verlinkten Ressourcen hinzugefügt werden sollen.

4 Klicken Sie auf die Schaltfläche „Erweitert“, um die erweiterten Optionen anzuzeigen.

5 Aktivieren Sie die Option „Link zu Datei im Dateisystem“. Klicken Sie auf die Schaltfläche „Variablen“.

6 Wählen Sie eine definierte Pfadvariable aus oder klicken Sie auf „Neu“, um eine Pfadvariable zu erstellen.

Wenn Sie eine definierte Pfadvariable ausgewählt haben, lesen Sie bei Schritt 9 weiter. Wenn Sie auf „Neu“ geklickt haben, wird das Dialogfeld „Neue Variable“ angezeigt.

7 Geben Sie einen Namen für die Pfadvariable ein und geben Sie dann den Pfad zur Datei bzw. zum Ordner an (oder klicken Sie auf „Datei“ bzw. „Ordner“ um die Datei bzw. den Ordner auszuwählen.

Klicken Sie auf „OK“, um die Pfadvariable zu erstellen.

8 Wählen Sie die neue Pfadvariable im Dialogfeld „Pfadvariable auswählen“ aus und klicken Sie auf „OK“.

9 Klicken Sie auf „Fertig stellen“, um die Ressource zu verlinken.

Pfadvariablen können auch über die Workbench-Voreinstellungen in Flash Builder definiert und verwaltet werden: Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“ > „Verlinkte Ressourcen“.

Ressourcen zwischen Projekten in einem Arbeitsbereich verschieben

Ressourcen verschieben

Wenn Sie mit mehreren Projekten in einem Arbeitsbereich arbeiten, können Sie Ressourcen von einem Projekt in ein anderes verschieben.

1 Wählen Sie im Paket-Explorer die Ressource aus, die verschoben werden soll.

2 Führen Sie einen der folgenden Schritte aus:

- Ziehen Sie die Ressource auf ein neues Projekt.
- Schneiden Sie die Ressource aus und fügen Sie sie in ein anderes Projekt ein.

Beim Verschieben von Ressourcen zwischen Projekten können Sie festlegen, ob Verweise aktualisiert werden sollen.

Hinweis: *Sie können sowohl normale als auch verlinkte Ressourcen verschieben. Informationen zum Verlinken von Ressourcen finden Sie unter „Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken“ auf Seite 46.*

Ressourcen gemeinsam nutzen

Um Ressourcen in mehreren Projekten zu nutzen, platzieren Sie alle gemeinsam genutzten Ressourcen in Ordner, die dann über den Quellpfad des Projekts mit jedem Projekt verlinkt werden können. Dies ist die beste Methode zum Verwenden gemeinsam genutzter Ressourcen wie z. B. Klassen, MXML-Komponenten und Bilder. Durch sie wird gewährleistet, dass in allen Projekten immer die neuesten Ressourcen zur Verfügung stehen. Wenn Projekte kompiliert werden, werden die gemeinsam genutzten Ressourcen zur SWC-Datei hinzugefügt.

Externen Ressourcenordner zum Quellpfad hinzufügen

1 Wählen Sie das Projekt im Paket-Explorer aus.

2 Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“ (bzw. „ActionScript-Erstellungspfad“, wenn Sie mit einem ActionScript-Projekt arbeiten).

3 Wählen Sie auf der Eigenschaftenseite für den Erstellungspfad die Registerkarte „Quellpfad“.

4 Klicken Sie auf die Schaltfläche „Ordner hinzufügen“.

5 Geben Sie den Pfad zum Ordner ein oder klicken Sie auf „Durchsuchen“ und wählen Sie dann den Pfad aus. Klicken Sie anschließend auf „OK“.

   Der Ordner wird zum Quellpfad hinzugefügt.

Auf der Registerkarte „Quellpfad“ können Sie auch Elemente im Quellpfad bearbeiten, löschen oder ihre Reihenfolge ändern.

Ordner, die zum Quellpfad hinzugefügt wurden, sind im Paket-Explorer markiert.

Ressourcen im Arbeitsbereich aktualisieren

Wenn Sie Ressourcen eines Projekts bearbeiten, hinzufügen oder löschen, aktualisiert die Workbench automatisch die verschiedenen Ansichten, die diese Ressourcen anzeigen. Wenn Sie zum Beispiel eine Datei aus dem Projekt löschen, wird diese Änderung sofort im Paket-Explorer angezeigt.

Sie können auch Ressourcen außerhalb von Flash Builder direkt im Dateisystem bearbeiten. Diese Änderungen sind in Flash Builder erst sichtbar, nachdem Sie den Arbeitsbereich aktualisiert haben.

In der eigenständigen Konfiguration von Flash Builder wird der Arbeitsbereich standardmäßig automatisch aktualisiert. Diese Option ist in den Flash Builder-Voreinstellungen konfigurierbar. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“. Sie können auch das Standardverhalten von Flash Builder ändern, wenn der Arbeitsbereich nie automatisch aktualisiert werden soll.

Arbeitsbereich manuell aktualisieren

❖ Wählen Sie im Paket-Explorer im Kontextmenü „Aktualisieren“. Alle Projektressourcen im Arbeitsbereich werden aktualisiert.

Automatische Aktualisierungsoption deaktivieren

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“.

2 Deaktivieren Sie die Option „Automatisch aktualisieren“.

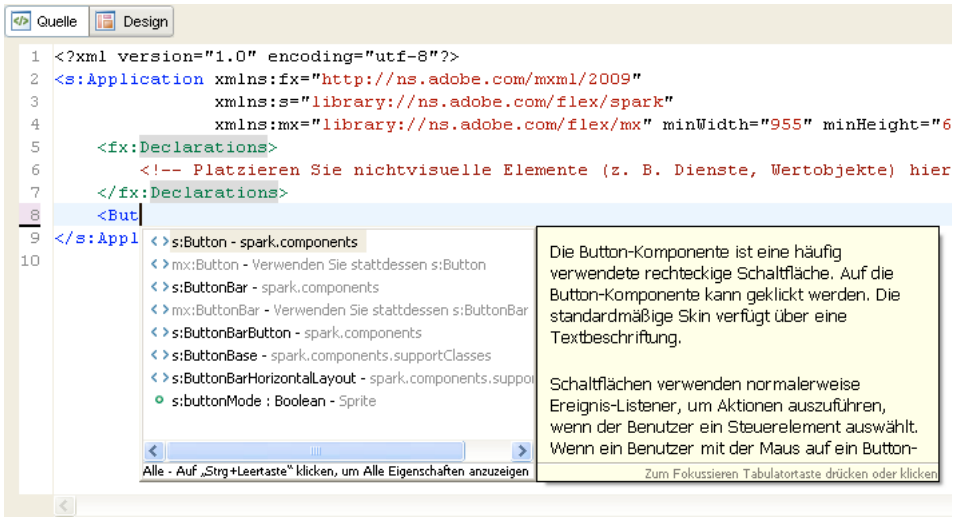
Kapitel 3: Tools zur Codeentwicklung in Flash Builder

In Adobe® Flash® Builder™ bearbeiten Sie MXML-, ActionScript- und CSS-Code in separaten Editoren. Die Flash Builder-Workbench ist sowohl projekt- als auch dokumentorientiert. Der entsprechende Editor wird automatisch geöffnet, da die Editoren mit Ressourcentypen verknüpft sind. Alle Editoren in Flash Builder verfügen über Codehinweise, Navigations-, Formatierungs-, Refaktorierungs- und andere produktivitätssteigernde Funktionen.

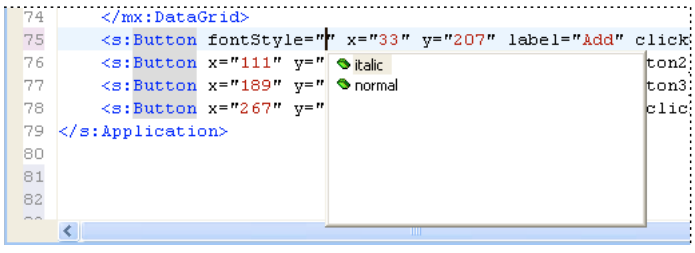
Inhaltshilfe

Wenn Sie MXML-, ActionScript- und CSS-Code eingeben, werden Hinweise und ASDoc-Referenzdokumentation angezeigt, um Ihnen beim Vervollständigen des Codes zu helfen. Diese Funktion heißt *Inhaltshilfe*.

Wenn Sie im MXML-Editor in einer MXML-Komponente schreiben, wird eine Liste mit allen Eigenschaften für diese Komponente angezeigt. Das folgende Beispiel zeigt Codehinweise für Eigenschaften einer MXML-Komponente:


Quelle
Design
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/mx" minWidth="955" minHeight="6
<fx:Declarations>
<!-- Platzieren Sie nichtvisuelle Elemente (z. B. Dienste, Wertobjekte) hier
</fx:Declarations>
<But
</s:Appl
s:Button - spark.components
mx:Button - Verwenden Sie stattdessen s:Button
s:ButtonBar - spark.components
mx:ButtonBar - Verwenden Sie stattdessen s:ButtonBar
s:ButtonBarButton - spark.components
s:ButtonBase - spark.components.supportClasses
s:ButtonBarHorizontalLayout - spark.components.suppor
s:buttonMode : Boolean - Sprite
Die Button-Komponente ist eine häufig verwendete rechteckige Schaltfläche. Auf die Button-Komponente kann geklickt werden. Die standardmäßige Skin verfügt über eine Textbeschriftung.
Schaltflächen verwenden normalerweise Ereignis-Listener, um Aktionen auszuführen, wenn der Benutzer ein Steuerelement auswählt. Wenn ein Benutzer mit der Maus auf ein Button-
Alle - Auf „Strg+Leertaste" klicken, um Alle Eigenschaften anzuzeigen
Zum Fokussieren Tabulatortaste drücken oder klicken

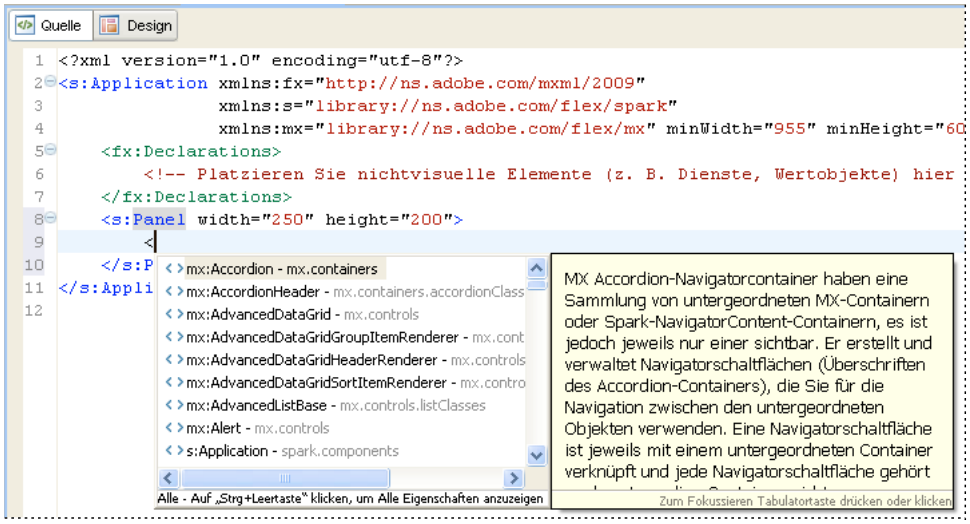
Wenn Sie Eigenschaften auswählen und eingeben, werden mögliche Eigenschaftswerte (sofern vordefinierte Werte existieren) angezeigt. Das folgende Beispiel zeigt Codehinweise für Eigenschaftswerte:



Die Inhaltshilfe funktioniert im ActionScript-Editor und im CSS-Editor ähnlich.

Inhaltshilfe im MXML-Editor

Im MXML-Editor werden Codehinweise automatisch angezeigt, während Sie Code eingeben. Das folgende Beispiel zeigt die Codehinweise, die angezeigt werden, wenn Sie einem Panel-Tag ein Tag hinzufügen. Außerdem wird die ASDoc-Referenzdokumentation angezeigt. Klicken Sie in den ASDoc-Inhalt oder drücken Sie F2, um den Inhalt in einem separaten Fenster mit Bildlaufmöglichkeit anzuzeigen. Klicken Sie außerhalb des ASDoc-Fensters, um es zu schließen.


Quelle
Design
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/mx" minWidth="955" minHeight="60
<fx:Declarations>
<!-- Platzieren Sie nichtvisuelle Elemente (z. B. Dienste, Wertobjekte) hier
</fx:Declarations>
<s:Panel width="250" height="200">
mx:Accordion - mx.containers
mx:AccordionHeader - mx.containers.accordionClass
mx:AdvancedDataGrid - mx.controls
mx:AdvancedDataGridGroupItemRenderer - mx.cont
mx:AdvancedDataGridHeaderRenderer - mx.controls
mx:AdvancedDataGridSortItemRenderer - mx.contro
mx:AdvancedListBase - mx.controls.listClasses
mx:Alert - mx.controls
s:Application - spark.components
MX Accordion-Navigatorcontainer haben eine Sammlung von untergeordneten MX-Containern oder Spark-NavigatorContent-Containern, es ist jedoch jeweils nur einer sichtbar. Er erstellt und verwaltet Navigatorschaltflächen (Überschriften des Accordion-Containers), die Sie für die Navigation zwischen den untergeordneten Objekten verwenden. Eine Navigatorschaltfläche ist jeweils mit einem untergeordneten Container verknüpft und jede Navigatorschaltfläche gehört
Alle - Auf „Strg+Leertaste" klicken, um Alle Eigenschaften anzuzeigen
Zum Fokussieren Tabulatortaste drücken oder klicken

Die Inhaltshilfe kategorisiert Codehinweise nach Typ, sodass sowohl visuelle als auch nicht visuelle MXML-Komponenten, Eigenschaften, Ereignisse und Stile angezeigt werden.

Standardmäßig zeigt die Inhaltshilfe ausschließlich Codehinweise für *empfohlene Typen* an. Empfohlene Typen sind Komponenten, die im deklarierten Namespace verfügbar sind oder auf andere Weise zu Verfügung stehen, je nach den einschließenden Tags. Die in einer Anwendung verfügbaren Komponenten richten sich nach der Namespace-Deklaration der Anwendung und den Tags, die den Einfügungspunkt im Editor einschließen.

Beispielsweise werden unter bestimmten Bedingungen ausschließlich Spark-Komponenten zugelassen. In anderem Kontext werden sowohl Spark- als auch Halo-Komponenten zugelassen. Von der Inhaltshilfe werden die Codehinweise entsprechend dem Kontext gefiltert.

Drücken Sie mehrmals Strg/Befehl+Leertaste, um wie unten aufgeführt zwischen den Filtern für angezeigte Codehinweise zu wechseln:

- Erste Anzeige: „Empfohlene Typen"
- Alle Komponenten
- Eigenschaften
- Ereignisse
- Effekte
- Stile
- Zurück zu „Empfohlene Typen"

Der Benutzer legt fest, welche Codehinweise und in welcher Reihenfolge diese angezeigt werden. Informationen zum Ändern der Standardeinstellung im Dialogfeld „Voreinstellungen" finden Sie unter „MXML-Code" auf Seite 272.

Inhaltshilfe im ActionScript-Editor

Bei der Eingabe von ActionScript-Code werden im ActionScript-Editor automatisch Codehinweise eingeblendet.

Von der Inhaltshilfe werden die Codehinweise entsprechend dem Kontext gefiltert. Drücken Sie mehrmals Strg/Befehl+Leertaste, um wie unten aufgeführt zwischen den Filtern für angezeigte Codehinweise zu wechseln:

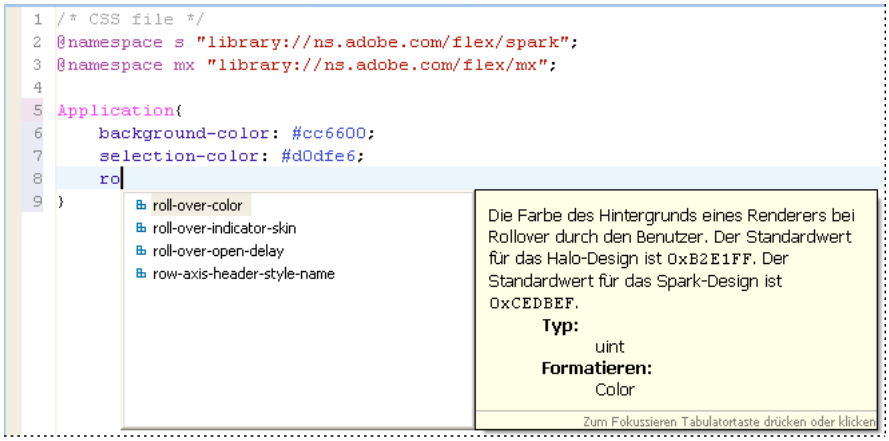
- Vorlagen
- Variablen
- Funktionen
- Klassen und Schnittstellen
- Pakete
- Namespaces

Der Benutzer legt fest, welche Codehinweise und in welcher Reihenfolge diese angezeigt werden. Informationen zum Ändern der Standardeinstellung im Dialogfeld „Voreinstellungen“ finden Sie unter „Code in ActionScript“ auf Seite 271.

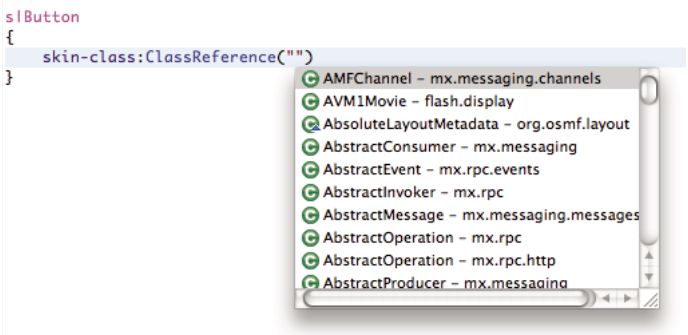
Inhaltshilfe im CSS-Editor

Die Inhaltshilfe zeigt Hinweise für CSS-Stile in eingebetteten `<fx:Style>`-Tags und in eigenständigen CSS-Dokumenten an, wie das folgende Beispiel zeigt:

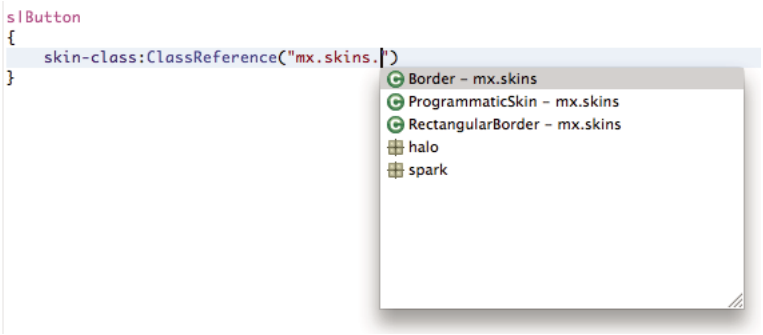
Hinweis: *Codehinweise in CSS-Dokumenten werden erst angezeigt, wenn Sie Strg/Befehl+Leertaste drücken.*



Die Inhaltshilfe enthält außerdem Hinweise zu Komponentenklassen in der ClassReference in einem CSS-Dokument, wie im folgenden Beispiel gezeigt:



Sie können auch den vollständig qualifizierten Klassennamen eingeben. In diesem Fall werden die verfügbaren Klassen und Pakete wie folgt angezeigt:



Das ClassReference-Tag ermöglicht die Navigation über Hyperlinks, sodass Sie zur referenzierten Komponente oder Skinklasse navigieren können. Drücken Sie dazu die Strg-Taste bzw. die Befehlstaste (Mac) und ziehen Sie den Cursor über das ClassReference-Tag. Der Klassenname wird dadurch zu einem Hyperlink. Weitere Informationen zur Navigation über Hyperlinks finden Sie unter „Codedefinitionen öffnen“ auf Seite 81.

ActionScript-Referenzübersicht anzeigen

1 Beginnen Sie, eine Codezeile einzugeben, die entweder eine MXML- oder eine ActionScript-Klasse enthält. Sie können auch mit der Maus auf die Klasse zeigen.

   Während Sie die Codezeile eingeben, wird neben den Codehinweisen die ActionScript-Referenzübersicht für die Klasse angezeigt. Wenn Sie mit der Maus auf eine Klasse zeigen, wird nur die ActionScript-Referenzübersicht angezeigt.

2 Klicken Sie in die ActionScript-Referenzübersicht oder drücken Sie F2, um die ActionScript-Referenzübersicht in einem separaten Fenster mit Bildlaufmöglichkeit anzuzeigen. Wenn Sie die Dokumentation gelesen haben, klicken Sie auf eine beliebige Stelle außerhalb des Fensters. Das ASDoc-Fenster wird geschlossen.

3 Zum Aufrufen der ASDoc-Ansicht drücken Sie Strg+3, geben „asdoc“ ein und wählen „Ansichten“:

Schnellhilfe

Die Schnellhilfe in Adobe Flash Builder bietet kontextbezogene Unterstützung, um die schnelle Erledigung von Aufgaben zu erleichtern. In der Schnellhilfe können Sie in einer Liste mit für Ihr aktuelles Codefragment relevanten Aktionen die gewünschte Aktion wählen.

Die Schnellhilfe rufen Sie über das Kontextmenü des Editors auf. Anschließend wählen Sie die benötigte Aktion. Sie können dazu auch die Tastenkombination Strg+1 (Windows) oder Befehlstaste+1 (Mac OS) verwenden.

Derzeit bietet die Schnellhilfe die folgenden Optionen:

In Datei umbenennen
Sie können die Schnellhilfe zum Umbenennen von Codeelementen verwenden, bei denen die Aktion „Umbenennen/Refaktorieren“ aktiviert ist. Dazu gehören Variablen, Methoden, Klassennamen, Parameter, Importe sowie Status und ActionScript-Code innerhalb von MXML-Tags.

Um alle Instanzen einer Variablen oder Methode in der Datei umzubenennen, platzieren Sie den Cursor im ausgewählten Variablen- oder Methodennamen und drücken Sie Strg+1. Wählen Sie dann die Option „In Datei umbenennen“, um die Variable oder Methode umzubenennen. In ähnlicher Weise können Sie die ID-Eigenschaft von MXML-Komponenten ändern.

In Arbeitsbereich umbenennen
Über die Schnellhilfe können Sie Codeelemente, bei denen die Aktion „Umbenennen/Refaktorieren“ aktiviert ist, in mehreren Dateien in Ihrem Arbeitsbereich umbenennen.

Um alle Instanzen einer Variablen oder Methode im Arbeitsbereich umzubenennen, platzieren Sie den Cursor im ausgewählten Variablen- oder Methodennamen und drücken Sie Strg+1. Wählen Sie dann die Option „In Arbeitsbereich umbenennen“, um die Variable oder Methode umzubenennen. Anschließend können Sie alle Referenzen auf die Variable oder Methode im Arbeitsbereich aktualisieren.

Importe verwalten
Platzieren Sie den Cursor in der gewünschten Importanweisung und drücken Sie Strg+1. Wählen Sie dann die benötigte Option zum Strukturieren von Importen. Weitere Informationen zum Verwalten von Importanweisungen finden Sie unter „Importanweisungen organisieren“ auf Seite 88.

Importanweisungen generieren
Wenn eine undefinierte Variable vorhanden ist, können Sie den Cursor an einer beliebigen Stelle in der Codezeile platzieren und Strg+1 drücken. Eine Option zum Importieren der Komponente wird verfügbar. Wenn für die Komponente MX- und Spark-Entsprechungen vorhanden sind, werden beide Optionen angezeigt.

Der Code könnte z. B. die undefinierte Variable `btn` enthalten:

```
var btn:Button;
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile und drücken Sie Strg+1. Optionen zum Importieren der Button-Komponente werden angezeigt. Wenn Sie die Spark-Button-Komponente wählen, wird die folgende Importanweisung erstellt:

```
import spark.components.Button;
```

Variablendeklaration teilen
Die Schnellhilfe ermöglicht es, eine Variable in zwei Teile zu teilen: die Variablendeklaration und die Variableninitialisierung.

Die Schnellhilfe wird in den folgenden Situationen angezeigt.

Kontext	**Beispiel**

Lokale Variablen innerhalb einer Funktion	Angenommen, eine Funktion lautet wie folgt:
	`public function TestAS()`
	`{`
	`var i:int=10;`
	`}`
	Um die Variable mithilfe der Schnellhilfe zu teilen, platzieren Sie den Cursor an einer beliebigen Stellen in der Variablendeklaration `var i:int=10;` und drücken Sie Strg+1. Wählen Sie dann die Option zum Teilen der Variablendeklaration.
	Die Variable wird wie folgt geteilt:
	`public function TestAS()`
	`{`
	`var i:int;`
	`i=10``;`
	`}`

Mehrere Variablen innerhalb einer Funktion	Angenommen, eine Funktion lautet wie folgt:
	`public function TestAS()`
	`{`
	`var i:int=10, j:int=20;`
	`}`
	Um die Variablen mithilfe der Schnellhilfe zu teilen, platzieren Sie den Cursor an einer beliebigen Stellen in der Variablendeklaration `var i:int=10, j:int=20;` und drücken Sie Strg+1. Wählen Sie dann die Option zum Teilen der Variablendeklaration.
	Sie können die Variable wie folgt teilen:
	`public function TestAS()`
	`{`
	`var i:int, j:int=20;`
	`i=10;`
	`}`
	Sie können außerdem die Variable „j:int=20;“ teilen, indem Sie den Cursor an einer beliebigen Stelle in der Variablendeklaration platzieren und Strg+1 drücken. Wählen Sie dann die Option zum Teilen der Variablendeklaration.
	Die Variable wird wie folgt geteilt:
	`public function TestAS()`
	`{`
	`var i:int, j:int;`
	`j=20;`
	`i=10;`
	`}`

Variable zuweisen

Wenn Sie einen ActionScript-Ausdruck prüfen und dieser einen Wert zurückgibt, können Sie die Schnellhilfe aufrufen und eine Variable für diesen Ausdruck erstellen. Die Schnellhilfe steht nicht für Methoden zur Verfügung, die keinen oder den Rückgabetyp `void` haben.

Der Name der erstellten Variablen wird aus dem Namen der Funktion oder der ID im Ausdruck abgeleitet. Wenn der abgeleitete Name bereits vorhanden ist, werden hinzukommende Variablen aufsteigend nummeriert.

Angenommen, der Code lautet wie folgt:

```
var i:int;
i;
```

Platzieren Sie den Cursor nach dem `i` und drücken Sie Strg+1. Wählen Sie dann „Neuer lokaler Variable eine Anweisung zuweisen“. Eine Variable mit dem Namen `i2` wird erstellt. Da die Variable `i` bereits vorhanden ist, erhält die neue Variable wie folgt den inkrementierten Namen `i2`:

```
var i:int;
var i2:int = i;
```

Die Option „Variabler zuweisen“ wird z. B. in den folgenden Fällen angezeigt:

Kontext	**Beispiel**

Literalausdruck	Die Schnellhilfe unterstützt ein breites Spektrum einfacher und komplexer Ausdrücke.
	Angenommen, der Ausdruck lautet wie folgt:
	`100+150`
	Platzieren Sie den Cursor an einer beliebigen Stelle im Ausdruck und drücken Sie Strg+1. Sie erhalten Code wie den folgenden:
	`var number2:Number = 110 + 500;`

Methodenaufruf	Angenommen, in einer Methode ist der folgende Code enthalten:
	`var ac:ArrayCollection = new ArrayCollection();`
	`ac.createCursor();`
	Platzieren Sie den Cursor an einer beliebigen Stelle in `ac.createCursor();` und drücken Sie Strg+1. Wählen Sie dann die Option zum Zuweisen einer Anweisung zu einer neuen lokalen Variablen. Die lokale Variable wird wie folgt erstellt:
	`var createCursor:IViewCursor = ac.createCursor();`

Eigenschaftenzugriff	Angenommen, in einer Eigenschaft ist der folgende Code enthalten:
	`var ac:ArrayCollection = new ArrayCollection();`
	`ac.source;`
	Platzieren Sie den Cursor an einer beliebigen Stelle in `ac.source` und drücken Sie Strg+1. Wählen Sie dann die Option zum Zuweisen einer Anweisung zu einer neuen lokalen Variablen. Die lokale Variable wird wie folgt erstellt:
	`var source:Array = ac.source;`

Lokale Variable in Feld konvertieren

Wenn lokale Variablen in einer Funktion enthalten sind, können Sie mithilfe der Schnellhilfe ein Feld in der Klasse erstellen.

Angenommen, in einer Funktion ist wie folgt eine Variable enthalten:

```
var i:int = 10;
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Variablendefinition und drücken Sie Strg+1. Wählen Sie dann „Lokale Variable in Feld konvertieren“. Das Klassenfeld wird wie folgt erstellt:

```
private var i:int;
```

Das neue Feld erhält denselben Namen wie die lokale Variable, vorausgesetzt, es kommt nicht zu Konflikten auf Klassenebene. Wenn der Name bereits vorhanden ist, wird dem Feldnamen die nächsthöhere Nummer angehängt und Sie können die Variable oder Methode in der Datei umbenennen.

Getter/Setter generieren

Die Schnellhilfe bietet die Möglichkeit zum Generieren von Gettern und Settern (Get- und Set-Accessor-Funktionen) für Klassenvariablen.

Angenommen, Sie arbeiten mit Code wie dem folgenden:

```
private var result:ResultEvent;
```

Platzieren Sie den Cursor in der Variablen `result` und drücken Sie Strg+1. Wählen Sie dann die Schnellhilfeoption zum Erstellen des Getters und Setters für „result“. Das Dialogfeld „Getter/Setter generieren“ wird angezeigt. In diesem Dialogfeld können Sie eine neue Getter- und Setter-Funktion festlegen. Weitere Informationen zum Generieren von Get- und Set-Funktionen finden Sie unter „Get-/Set-Accessor-Funktionen generieren“ auf Seite 74.

Funktionen generieren

Die Schnellhilfe bietet die Möglichkeit zum Generieren spezieller Funktionen wie labelFunction, iconFunction usw.

Im folgenden Beispiel soll eine labelFunction für den folgenden Code erstellt werden:

```
<mx:DataGrid labelFunction="lblfunc" dataTipFunction="tipFunc" />
```

Platzieren Sie den Cursor in `"lblfunc"` und drücken Sie Strg+1. Wählen Sie dann die Schnellhilfeoption zum Erstellen der labelFunction. Für die Funktion wird wie folgt Stubcode erstellt:

```
protected function lblfunc(item:Object, column:DataGridColumn):String
    {
    // TODO Auto-generated method stub
    }
```

Ereignisprozeduren aus MXML-Code generieren

Die Schnellhilfe ermöglicht das Generieren von Ereignisprozeduren mit benutzerdefinierten Handlernamen innerhalb von MXML-Code.

Um z. B. eine Ereignisprozedur durch Klicken auf das <mx:Button>-Tag zu erstellen, geben Sie den folgenden Code ein:

```
<mx:Button click="clickHandler" />
```

Platzieren Sie anschließend den Cursor in `"clickHandler"`, drücken Sie Strg+1 und wählen Sie „Ereignisprozedur generieren“. Die Ereignisprozedur und der Stubcode werden wie folgt generiert:

```
<mx:Button click="clickHandler(event)" />

protected function clickHandler(event:MouseEvent):void
    {
        // TODO Auto-generated method stub
    }
```

Sie können auch eine Ereignisprozedur für den folgenden Code erstellen, indem Sie Strg+1 drücken, während sich der Cursor an einer beliebigen Stelle innerhalb von `"clickHandler(event)"` befindet.

```
<mx:Button click="clickHandler(event)" />
```

Wie Sie den von Flash Builder vordefinierten Stubcode anpassen, erfahren Sie unter „Codevorlagen“ auf Seite 57.

Methoden überschreiben oder implementieren

Flash Builder bietet eine Option zum Auswählen und Überschreiben von Methoden der übergeordneten Klasse oder zum Implementieren von Schnittstellenmethoden.

1 Öffnen Sie das Dialogfeld „Methoden überschreiben/implementieren“, indem Sie im Menü „Quelle“ die Option „Methoden überschreiben/implementieren“ wählen. Oder wählen Sie im Kontextmenü des MXML- oder ActionScript-Editors „Quelle“ > „Methoden überschreiben/implementieren“.

2 Die Methoden der jeweils übergeordneten Klasse werden in einer Baumstruktur angezeigt. Pro Klasse können Sie die Methoden auswählen, die überschrieben bzw. implementiert werden sollen.

3 Sie können die Einfügemarke wählen, an der die gewählten Methoden eingefügt werden sollen. Die standardmäßige Einfügemarkenoption hängt davon ab, an welcher Editorposition Sie den Cursor beim Öffnen des Dialogfelds „Methoden überschreiben/implementieren“ platziert hatten. Die der Cursorposition am nächsten gelegene Variable oder Methode wird als Einfügemarkenoption eingeblendet.

   Flash Builder generiert Stub-Code für die gewählten Methoden.

Wie Sie den von Flash Builder vordefinierten Stubcode anpassen, erfahren Sie unter „Codevorlagen“ auf Seite 57.

Codevorlagen

Mit Codevorlagen wird die Kodierung beschleunigt, indem Sie häufig verwendete Kodierungsmuster automatisch einfügen lassen.

Flash Builder weist eine Reihe vordefinierter Codevorlagen auf. Darüber hinaus können Sie zusätzliche Codevorlagen für häufig verwendete Codemuster verwenden. Um alle verfügbaren Codevorlagen anzuzeigen, öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie die Optionen „Flash Builder“ > „Editoren“ > „Codevorlagen“ aus.

Adobe Community-Experte Paul Robertson schrieb einen Artikel über die Verwendung von Codevorlagen.

MXML-, ActionScript- und CSS-Codevorlagen

Die Codevorlagen MXML, ActionScript und CSS sind kontextabhängig. Sie können durch Strg+Leertaste aufgerufen werden. Sie können die mit Flash Builder ausgelieferten vordefinierten Vorlagen übernehmen oder eigene Vorlagen erstellen.

Codevorlagen einfügen

Sie fügen eine Codevorlage in den Codeeditor ein, indem Sie den Namen der Vorlage in den Codeeditor eingeben und Strg/Befehl+Leertaste drücken.

Beispiel: Beim Schreiben von ActionScript-Code verwenden Sie wiederholt die `for`-Schleife. In diesem Fall definieren Sie eine Codevorlage für die `for`-Schleife wie folgt:

```
for (var i:int = 0; i < array.length; i++) { }
```

Wenn Sie eine Codevorlage verwenden, brauchen Sie nicht den gesamten Code der `for`-Schleife einzugeben. Geben Sie stattdessen in der ActionScript-Klasse '`for`' ein und drücken Sie Strg+Leertaste. Daraufhin wird eine Vorlagenoption zum Erstellen der `for`-Schleife angezeigt. Bei Auswahl der Codevorlage wird der in der Codevorlage definierte Code eingefügt.

Vorlagen können auch Vorlagenvariablen enthalten. Eine Vorlagenvariable wird in `${}` definiert. Die Variable wird gemäß der entsprechenden Variablendefinition im Editor aufgelöst.

Beispiel: Sie definieren eine Codevorlage für die `for`-Schleife wie folgt:

```
for (var ${index}:int = 0; ${index} < ${array}.length; ${index}++) { ${cursor} }
```

Anschließend rufen Sie die Codevorlage auf, nachdem Sie die Variable `myArr` wie folgt definiert haben:

```
{
    var myArr:ArrayCollection = null;
}
```

Dann wird `${array}` in der Codevorlage zu`myArr` aufgelöst. Der entsprechende Code sieht so aus:

```
{
    var myArr:ArrayCollection = null;
    for (var ${index}:int = 0; ${index} < myArr.length; ${index}++) { ${cursor} }
}
```

Codevorlagen erstellen und bearbeiten

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie die Optionen „Flash Builder“ > „Editoren“ > „Codevorlagen“.

2 Codevorlagen werden nach ActionScript-, MXML- und CSS-Codevorlagen kategorisiert. Unter jeder Kategorie finden Sie eine Reihe vordefinierter Codevorlagen. Sie können eine bestehende Vorlage bearbeiten oder eine neue hinzufügen.

3 Fügen Sie eine neue Vorlage hinzu, indem Sie die Kategorie der Codevorlage auswählen und dann auf „Hinzufügen“ klicken. Geben Sie im Dialogfeld „Neue Vorlage“ einen Namen für die Codevorlage und eine Kurzbeschreibung ein. Geben Sie dann den Kontext an, in dem die Codevorlage aufgerufen werden soll.

   Für ActionScript- und MXML-Codevorlagen ist die Angabe des Kontexts möglich.

   Sie können aus den folgenden ActionScript-Kontextoptionen auswählen:

   - **ActionScript**: Fügt die Codevorlage an beliebiger Stelle des ActionScript-Dokuments ein
   - **ActionScript statement**: Fügt die Codevorlage in Funktionen sowie in Elemente einer Klasse ein
   - **ActionScript members**: Fügt die Codevorlage nur in Elementen einer Klasse ein
   - **ActionScript Package Scope**: Fügt die Codevorlage wie folgt in einem Paket ein:

     ```
     Package
         {
         /* insert code template*/
         }
     ```

   Sie können aus den folgenden MXML-Kontextoptionen auswählen:

   - **MXML**: Fügt die Codevorlage an beliebiger Stelle des MXML-Dokuments ein
   - **MX Component**: Fügt die Codevorlage in MX-Komponenten, die für das Flex 3 SDK verfügbar sind, ein
   - **Spark Components**: Fügt die Codevorlage in Spark-Komponenten, die für das Flex 4 SDK verfügbar sind, ein
   - **MXML attributes**: Fügt die Codevorlage für MXML-Attribute in MX- und Spark-Komponenten ein.

4 Geben Sie den Code für die Vorlage im Bereich „Muster“ ein. Um Variablen in den Code einzufügen, klicken Sie auf „Variable einfügen“ und treffen Sie eine Auswahl aus einer Liste vordefinierter Variablen. Die Variablen beziehen sich auf den Kontext der Vorlagenkategorie.

   ActionScript-Vorlagen enthalten vordefinierte Variablen wie `array`, `enclosing_method`, `enclosing_package`, `enclosing_type`, `field`, `local_var` und `var`. MXML-Vorlagen enthalten vordefinierte Variablen wie `fx`, `mx`, `s` und `tag`.

5 Wenn Flash Builder die Codevorlage nicht automatisch in den Code einfügen soll, deaktivieren Sie die Option „Automatisch in Code einfügen“.

6 Um eine Codevorlage anzupassen, wählen Sie die Vorlage und klicken Sie auf „Bearbeiten“. Klicken Sie nach Bearbeitung der Vorlage auf „OK“.

   Weitere Informationen zum Anpassen von Dateivorlagen und Vorlagenvariablen finden Sie unter „Dateivorlagen anpassen“ auf Seite 63 und „Vorlagenvariablen“ auf Seite 66.

Sie können die angepasste Vorlage jederzeit entfernen und die vordefinierte Codevorlage wiederherstellen, indem Sie auf „Auf Standardwert zurücksetzen“ klicken.

Die vordefinierten Vorlagen können Sie standardmäßig über die Inhaltshilfe aufrufen. Wenn jedoch eine bestimmte Vorlage nicht in den Optionen der Inhaltshilfe eingeblendet werden soll, deaktivieren Sie diese Vorlage im Abschnitt „Bestehende Vorlagen“.

Sie können Codevorlagen auch importieren und exportieren. Wählen Sie dazu eine oder mehrere Vorlagen und exportieren Sie sie. Die Vorlagen werden als XML-Datei exportiert.

Adobe Community-Experte Paul Robertson schrieb einen Artikel über das Gemeinsame Nutzen von Codevorlagen.

Codevorlagen in Flash Builder

Flash Builder kann in folgenden Szenarien automatisch vordefinierten Code generieren:

- „Ereignisprozeduren generieren“ auf Seite 218
- „Get-/Set-Accessor-Funktionen generieren“ auf Seite 74
- „Aus der Verwendung generieren“ auf Seite 72 (Stubcode als Platzhalter für eine undefinierte Methode)
- „Methoden überschreiben oder implementieren“ auf Seite 56

Sie können die von Flash Builder generierte vordefinierte Codevorlage anpassen.

Codevorlage anpassen

1. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie die Optionen „Flash Builder“ > „Editoren“ > „Codevorlagen“ > „Flash Builder“.
2. Wählen Sie den Namen der anzupassenden Codevorlage aus und klicken Sie auf „Bearbeiten“. Um zum Beispiel den Code anzupassen, der bei der Generierung einer Ereignisprozedur erzeugt wird, wählen Sie die entsprechende Vorlage aus und klicken Sie auf „Bearbeiten“.
3. Sie können den Namen der Vorlage, die Beschreibung und das Codemuster anpassen.
4. Um eine Variable in den Code einzufügen, klicken Sie auf „Variable einfügen“ und wählen die gewünschte Variable aus. Weitere Informationen zu den verfügbaren Codevariablen finden Sie unter „Codevariablen verwenden“ auf Seite 60.
5. Ihre Änderungen können Sie jederzeit mit einem Klick auf „Standard wiederherstellen“ verwerfen.
6. Sie können die Codevorlage auch importieren und exportieren. Wählen Sie dazu eine oder mehrere Vorlagen und exportieren Sie sie. Die Vorlagen werden als XML-Datei exportiert.

Codevariablen verwenden

Codevariablen für Ereignisprozeduren

Variable	Beschreibung	Beispiel
${component_id}	Wird in der eindeutigen Komponenten-ID aufgelöst.	Lautet die ID der Schaltflächenkomponente `test`, so wird die Ereignisprozedur `test_clickHandler` erzeugt. Wenn Sie keinen ID-Wert angegeben haben, lauten die automatisch erzeugten Werte component1, component2 usw.
${component_name}	Wird im Namen des Tags aufgelöst.	`${namespace} ${modifiers}function ${:method_name('${component_id}_${event_name}Handler')}(${event}:${event_type}):${return_type} { // TODO Auto-generated method stub ${cursor} }`
${event_name}	Gibt den Namen des Ereignisses an.	`clickEvent`, `onHover`
${event_type}	Wird im Ereignisprozedurtyp aufgelöst.	Flash Builder bestimmt einen Standardereignistyp für jede Komponente der Benutzeroberfläche. Bei der Erzeugung eines Klickereignisses für eine Schaltflächenkomponente wird eine Ereignisprozedur des Typs MouseEvent wie folgt generiert: `button1_clickHandler(event:MouseEvent)`
${modifiers}	Legt die Modifizierer für die generierte Funktion fest.	`static`
${method_name}	Wird in der Ereignisprozedur aufgelöst.	Für ein Klickereignis der Schaltflächenkomponente kann der Name der Ereignisprozedur button1_clickHandler lauten.
${namespace}	Legt den Namespace-Wert für die generierte Funktion fest.	Folgende Namespace-Werte können vorkommen: • `protected` • `public` • `private` Der Standardwert ist `protected`.

Beispiel mit Codevariablen für Ereignisprozedurfunktionen:

```
${namespace} ${modifiers}function
${:method_name('${component_id}_${event_name}Handler')}
${event}:${event_type}):${return_type}
    {
        // TODO Auto-generated method stub
    ${cursor}
    }
```

Codevariablen für Get- und Set-Accessor-Funktionen

Variable	Beschreibung	Beispiel
${metadata}	Gibt die zu generierenden Metadaten-Tags an.	Generieren von Get- und Set-Accessor-Funktionen für eine Bindable-Variable

${asdoc}	Gibt das ASDoc-Dokumente an, die für die Get- und Set-Accessor-Funktionen erzeugt werden sollen.	`${metadata}`
		`${asdoc}${namespace}`
		`${modifiers}function get`
		`${method_name}()${return_type}`
		`{`
		`return ${property};`
		`}`

${return_type}	Wird im Variablentyp aufgelöst.
	Wenn der Variablentyp nicht angegeben ist, verfügt die erzeugte Get-Accessor-Funktion über keinen Rückgabetyp.
		`${metadata}`
		`${asdoc}${namespace}`
		`${modifiers}function get`
		`${method_name}()${return_type}`
		`{`
		`return ${property};`
		`}`

${property}}	Wird im Eigenschaftsnamen des Getter-/Setter-Dialogs aufgelöst.	Bei einer Variable `var i:int` wird i in `_i` aufgelöst.
${argument_type}	Wird im Datentyp der generierten Set-Funktion aufgelöst.	

Beispiel mit Codevariablen für Get- und Set-Accessor-Funktionen:

```
${asdoc}
${namespace} ${modifiers}function set ${method_name}(value${argument_type}):void
{
    if( ${property} !== value)
        {
            ${property} = value;
            dispatchEvent(new Event("${event_name}"));
        }
}
```

Codevariablen für Funktionen in einer undefinierten Klasse

Variable	Beschreibung	Beispiel
${params}	In einer undefinierten Funktion, die eine bestimmte Anzahl an Argumenten annehmen kann, gilt für die generierte Funktion die gleiche Anzahl an Argumenten wie für den Typ.	

Metadaten-Codevervollständigung

Flash Builder zeigt Hinweise zur Codevervollständigung für Metadaten an, die Sie in MXML- und ActionScript-Dokumenten verwenden.

In MXML-Dokumenten werden Codehinweise für Metadaten in eingebetteten `<fx:Metadata>`- und `<fx:Script>`-Tags angezeigt. In ActionScripts werden Codehinweise für Metadaten auch für Elemente der ActionScript-Sprache wie Klassennamen, Variablen, Getter und Setter angezeigt.

Die Codehinweise richten sich jeweils nach dem Kontext des MXML- und ActionScript-Dokuments und des Codes, in dem die Metadaten verwendet werden. Wenn Sie z. B. innerhalb zweier leerer Zeilen in einer ActionScript-Anweisung die Inhaltshilfe aufrufen, werden Codehinweise angezeigt, die nur für diese ActionScript-Anweisung gelten. Um alle für das ActionScript- oder MXML-Dokument verwendbaren Codehinweise anzuzeigen, drücken Sie mehrmals Strg+Leertaste. Dadurch werden die verfügbaren Codehinweise nacheinander angezeigt.

Metadaten-Codevervollständigung in MXML-Dokumenten verwenden

In MXML-Dokumenten oder -Klassen haben Sie die folgenden Möglichkeiten zum Verwenden der Codevervollständigung für Metadaten:

- Geben Sie „[“ in `<fx:Metadata>`-Tags eingeschlossen ein:

```
<fx:Metadata>
    [
</fx:Metadata>
```

- Geben Sie „[“ in `<fx:Script>`-Tags eingeschlossen ein:

```
<fx:Script>
    <![CDATA[
        [
    ]]>
</fx:Script>
```

Metadaten-Codevervollständigung in ActionScript-Dokumenten verwenden

In ActionScript-Dokumenten können Sie die Codevervollständigung für Metadaten verwenden, indem Sie vor einem Klassennamen, einer Variablen, einem Getter oder einem Setter ein „[“ eingeben:

```
[
    class GetSet
    {
        [
            private var privateProperty:String;
            [
            public function get publicAccess():String
            {
                return privateProperty;
            }
            [
            public function set publicAccess(setValue:String):void
            {
                privateProperty = setValue;
            }
}
```

Codevervollständigung für angepasste Metadatentags

Flash Builder unterstützt die Codevervollständigung für benutzerdefinierte Metadaten-Tags, die bei der Verwendung von Flex-Frameworks anderer Hersteller übernommen werden.

Um Codehinweise für benutzerdefinierte Metadaten-Tags in Ihrem Projekt zu aktivieren, generieren Sie wie folgt eine SWC-Datei, die eine Datei `metadata.xml` enthält:

1. Erstellen Sie ein Bibliotheksprojekt. Wenn Sie ein Bibliotheksprojekt erstellen, führt der Assistent „Neues Flex-Bibliotheksprojekt“ Sie durch die einzelnen Schritte und fordert Sie auf, den Projektnamen, den Pfad und Informationen zum Erstellungspfad anzugeben. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte erstellen“ auf Seite 235.
2. Fügen Sie die Datei `metadata.xml` im Ordner „src“ im Stammordner des Bibliotheksprojekts hinzu. Nehmen Sie alle gewünschten Metadaten-Tags in die Datei `metadata.xml` auf.

   Fügen Sie gegebenenfalls die Datei `metadata.properties` im Ordner für das benötigte Gebietsschema hinzu. Beispiel: locale/en_US oder locale/ja_JP.

   Weitere Informationen zu Metadaten-Tags finden Sie unter Metadata-Tags in der Dokumentation zu Flex.
3. Nehmen Sie die Datei `metadata.xml` wie folgt in die SWC-Datei für die Bibliothek auf:
   a. Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex Bibliothek-Erstellungspfad“.

      Die hinzugefügte Datei `metadata.xml` wird auf der Registerkarte „Elemente“ angezeigt.

   b. Wählen Sie die in die SWC-Datei aufzunehmende Datei `metadata.xml` aus und klicken Sie auf „OK“.

      Die SWC-Datei wird kompiliert und im Ausgabeordner (Bin) des Bibliotheksprojekts generiert.
4. Wählen Sie den Gebietsschemaordner aus, dem Sie die Datei `metadata.properties` (sofern vorhanden) hinzugefügt haben.
5. Nachdem Sie die SWD-Datei generiert haben, fügen Sie sie wie folgt dem Build-Pfad Ihres Projekts hinzu:
   1. Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.
   2. Klicken Sie auf „SWC hinzufügen“.
   3. Geben Sie den Speicherort der SWC-Datei ein oder navigieren Sie dorthin und klicken Sie auf „OK“.

Nachdem Sie die SWC-Datei dem Build-Pfad hinzugefügt haben, werden Codevervollständigungshinweise für Metadaten für die in der Datei `metadata.xml` definierten Metadaten-Tags angezeigt. Sie können die SWC-Datei für mehrere Anwendungen verwenden oder an andere Entwickler weitergeben.

Dateivorlagen anpassen

Mit Flash Builder können Sie die in neuen MXML-, ActionScript- und CSS-Dateien enthaltenen Standardinformationen anpassen. Beispiele für Informationen, die Sie angeben können, sind Variablen zur Angabe von Autor und Datum, Variablen für öffnende und schließende Tags und Attribute, Variablen für verschiedene ActionScript-Deklarationen, Namespace-Präfixe sowie so gut wie jeder Inhalt, den Sie in eine Vorlagendatei aufnehmen möchten. Dateivorlagen sind besonders praktisch, um einführende Kommentare und Copyrighthinweise anzugeben.

Der Inhalt einer neuen Datei wird in einer Dateivorlage festgelegt, die über „Voreinstellungen“ > „Flash Builder“ > „Dateivorlagen“ verfügbar ist. Vorlagen stehen für folgende Dateitypen zur Verfügung:

ActionScript	ActionScript-Datei ActionScript-Klasse ActionScript-Schnittstelle Skinfähige ActionScript-Komponente
MXML	MXML-Webanwendung MXML-Desktopanwendung MXML-Komponente MXML-Modul MXML-Skin ItemRenderer für Spark-Komponenten ItemRenderer für MX-Komponenten ItemRenderer für MX-DataGrid ItemRenderer für AdvancedDataGrid ItemRenderer für MX Tree
FlexUnit	FlexUnit-TestCase-Klasse FlexUnit-TestSuite-Klasse FlexUnit4-TestCase-Klasse FlexUnit4-TestSuite-Klasse
CSS	CSS-Datei.

Nachdem Sie eine Vorlage geändert haben, können Sie sie exportieren und gegebenenfalls an Kollegen weitergeben.

Dateivorlage ändern

1. Wählen Sie „Voreinstellungen“ > „Flash Builder“ > „Dateivorlagen“.
2. Erweitern Sie die Kategorie „Dateitypen“ und wählen Sie eine Dateivorlage aus, die geändert werden soll.
3. Klicken Sie auf „Bearbeiten“ und ändern Sie die Vorlage.

   Sie können direkt in den Vorlageneditor schreiben oder auf „Variablen einfügen“ klicken, um vordefinierte Daten in die Vorlage einzufügen.
4. Klicken Sie zum Speichern der Änderungen auf „OK“.

   Die Änderungen werden auf neue Dateien angewendet.

Dateivorlagen exportieren und importieren

1. Wählen Sie „Voreinstellungen“ > „Flash Builder“ > „Dateivorlagen“.
2. Erweitern Sie die Kategorie „Dateitypen“ und wählen Sie eine Dateivorlage aus.
3. Klicken Sie auf „Exportieren“, um eine Vorlage in das Dateisystem zu exportieren, bzw. auf „Importieren“, um eine zuvor exportierte Vorlage zu importieren.

   Vorlagen werden als XML-Dateien exportiert.

Standardwerte wiederherstellen

Hinweis: *Wenn Sie die Standardwerte wiederherstellen, werden alle Dateivorlagen auf die Standardeinstellungen zurückgesetzt. Es ist nicht möglich, nur eine Vorlage auf die Standardeinstellungen zurückzusetzen.*

- Um die Standardvorlagen wiederherzustellen, öffnen Sie „Voreinstellungen“ > „Flash Builder“ > „Dateivorlagen“ und klicken auf „Standardwerte wiederherstellen“.

Vorlagenvariablen

Vorlagenvariablen für alle Dateitypen

Variable	Beschreibung	Beispiel
${date}	Aktuelles Datum	15.2.2009
${year}	Aktuelles Jahr	2009
${time}	Aktuelle Zeit	15:15
${file_name}	Name der neu erstellten Datei	HalloWelt.mxml
${project_name}	Name des Flex- oder ActionScript-Projekts	Hallo_Welt_Projekt
${user}	Benutzername des Autors	hmustermann
$$ ${dollar}	Dollarsymbol	$

Vorlagenvariablen für MXML-Dateien

Variable	Beschreibung	Beispiel

${application}
${component}
${module}
	Legt den MXML-Tag-Namen der Anwendung, der Komponente oder des Moduls fest.
	Bei einer Webanwendung wird ${application} zu „Application“ erweitert.
	Bei einer Desktopanwendung wird ${application} zu „WindowedApplication“ erweitert.
	${component} wird zu „Component“ erweitert.
	${module} wird zu „Module“ erweitert.
	Diese Tags werden typischerweise verwendet, um das öffnende und schließende Tag einer Datei zu platzieren.
		Im folgenden Beispiel wird
		`<${application} ${xmlns}${wizard_attributes}${min_size}> ${wizard_tags}`
		`</${application}>`
		zu Folgendem erweitert:
		`<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009" xmlns:s="library://ns.adobe.com/flex/spark" xmlns:mx="library://ns.adobe.com/flex/halo" minWidth="1024" minHeight="768"> <s:layout> <s:BasicLayout/> </s:layout>`
		`</s:Application>`

${xml_tag}	XML-Version	`<?xml version="1.0" encoding="utf-8"?>`
${xmlns}	Löst die Namespace-Definition basierend auf dem Flex SDK-Typ des Projekts und dem Namespace-Präfix auf, die in den Voreinstellungen definiert sind.	Für ein Flex 4 SDK-Projekt: xmlns="http://ns.adobe.com/mxml/2009"
${min_size}	Mindestgröße einer MXML-Webanwendung.	`minWidth="1024" minHeight="768"`
${ns_prefix}	Namespace-Präfix für das Flex SDK des Projekts. Sie können die Standardwerte für diese Variable nicht ändern.	Flex 3: `mx:` Flex 4: `fx:`

Variable	Beschreibung	Beispiel
${wizard_attributes}	Legt die Position der Attribute fest, die über den Assistenten *Neue Datei* definiert wurden.	Für eine neue Webanwendung wird ${application} ${xmlns}${wizard_attributes}> zu Folgendem erweitert: <Application xmlns="http://ns.adobe.com/mxml/2009" layout="vertical">
${wizard_tags}	Legt die Layouteigenschaft für Container fest, die über den Assistenten *Neue Datei* definiert wurden.	Bei einer neuen Anwendung, die das Flex 4 SDK verwendet: <s:layout> <s:BasicLayout/> </s:layout>

${fx}	Präfix für den Dokument-Namespace in MXML 2009. Das Präfix entspricht der Angabe im MXML-Dokument.	Verwendung der folgenden Library-Tag-Vorlage in einem MXML-Dokument:
		`<${fx}Library>`
		`  <${fx}Definition id="${def}">`
		`  ${cursor}`
		`  </${fx}Definition>`
		`</${fx}Library>`
		Es wird wie folgt ein Bibliotheks-Tag erstellt:
		`<fxLibrary> <fxDefinition id="def">`
		`</fxDefinition> </fxLibrary>`

Variable	Beschreibung	Beispiel
${mx}	Präfix für den Namespace für MX-Dokumente. Das Präfix entspricht der Angabe im MXML-Dokument.	Verwendung der folgenden Combobox-Vorlage im MX-Komponenten-Kontext: `<${mx}ComboBox id="${comboBox}" rowCount="${rowCount:values(5)}" dataProvider="${dataProvider}"/> ${cursor}` Es wird wie folgt ein Kombinationsfeld erstellt: <mx:ComboBox id="comboBox" rowCount="5" dataProvider="dataProvider"/>

${s}	Präfix für den Namespace für Spark-Dokumente.	Verwendung der Spark-Button-Vorlage im Spark-Komponenten-Kontext:
		`<${s}Button id="${btn}" label="${myButton}" click="${onClick}(${event})"/> ${cursor}`
		Es wird wie folgt eine Spark-Schaltfläche erstellt:
		`<s:Button id="btn" label="myButton" click="onClick(event)"/>`

${ist}	Der vollständig qualifizierte Tag-Name für die MX-Komponenten des Projekts.	Verwendung der List-Vorlage im MXML-Kontext. `<${list:tag(mx.controls.List)} id="${myList}"> <${dp:tag(dataProvider)}> <${arraycollection:tag(mx.collections.ArrayCollection)}> ${cursor} </${arraycollection}> </${dp}> </${list}>` Es wird wie folgt eine Liste erstellt: `<s:List id="myList"> <s:dataProvider> <s:ArrayCollection> </s:ArrayCollection> </s:dataProvider> </s:List>`

Vorlagenvariablen für ActionScript-Dateien

Variable	Beschreibung	Beispiel
${package_declaration}	Generiert die Paketdeklaration.	Für eine Datei im Paket „com/samples“ wird Folgendes generiert: package com.samples
${import_declaration}	Generiert die erforderlichen Importdeklarationen für eine neue ActionScript-Klasse oder ActionScript-Schnittstelle.	Für eine Subklasse von TextBox wird Folgendes generiert: import flex.graphics.TextBox;
${interface_declaration}	Generiert die Schnittstellendeklaration für eine neue ActionScript-Schnittstelle.	Für eine neue Schnittstelle, die die IButton-Schnittstelle erweitert, wird Folgendes generiert: public interface IMyButton extends IButton
${class_declaration}	Generiert die Klassendeklaration für eine neue ActionScript-Klasse.	Für eine neue Subklasse von CheckBox wird Folgendes generiert: public class MyCheckBox extends CheckBox
${class_body}	Generiert alle erforderlichen Anweisungen für eine neue Klasse.	Für eine neue Subklasse von Button, die die IBorder-Schnittstelle implementiert, wird Folgendes für den Class Body generiert: `public function MyButton() { super(); } public function get borderMetrics():EdgeMetrics { return null; }`

${interface_name}
${class_name}
${package_name}
	Legt den Namen der Schnittstelle, Klasse oder des Pakets fest.
	Wird typischerweise beim Generieren von Kommentaren verwendet.
		Die Vorlagenspezifikation
		`/* * ${class_name} implements. . . */`
		generiert z. B. den folgenden Code:
		`/* * MyButton implements. . . */`

${array}	Gibt den Wert eines Arrays an.	Die folgende `Fore`-Vorlage:
		`for each (var ${index}:${type} in ${array}) { ${line_selection} ${cursor} }`
		enthält eine Iteration über den Wert eines Arrays unter Verwendung einer Aufzählung:
		`for each (var i:type in array) { }`

${enclosing_method}	Gibt den Namen der einschließenden Methode an.	Die `traceMethod`-Vorlage verfolgt die Methode wie folgt: `trace("${enclosing_type}.${enclosing_method} ${enclosing_method_arguments})");`

Variable	Beschreibung	Beispiel

${enclosing_package}	Gibt den Paketnamen an, z. B. „xx.yy“ in „xx.yy.class“.	Die `package`-Vorlage erstellt ein Paket wie folgt:
		`package ${enclosing_package}`
		`{`
		`/**`
		`* @author ${user}`
		`*/`
		`class ${enclosing_type}`
		`{`
		`${cursor}`
		`}`
		`}`

${enclosing_type}	Gibt den Typennamen an, z. B. „class“ in „xx.yy.class“.	Die `package`-Vorlage erstellt ein Paket, wobei der Klassenname wie folgt angegeben wird:
		`package ${enclosing_package}`
		`{`
		`/**`
		`* @author ${user}`
		`*/`
		`class ${enclosing_type}`
		`{`
		`${cursor}`
		`}`
		`}`

${field}	Gibt die Klassenvariablen an.	Die `do`-Vorlage erstellt wie folgt die do-while-Schleife:
		`do`
		`{`
		`${line_selection}`
		`${cursor}`
		`} while`
		`(${condition:local_var(Boolean)});`

${local_var}	Gibt die lokale Variable in der Blockebene an.	Die `if`-Vorlage erstellt wie folgt eine if-Anweisung:
		`if (${condition:local_var(Boolean)})`
		`{`
		`${cursor}`
		`}`

${var}	Gibt alle sichtbaren Variablen an.	Die `fori`-Vorlage iteriert wie folgt über den Wert des Arrays:
		`for (var ${index}:int = 0; ${index} < ${array}.length; ${index}++)`
		`{`
		`${cursor}`
		`}`

Vorlagenvariablen für CSS-Dateien

Variable	Beschreibung	Beispiel

${css_namespaces}	Definiert Namespaces für Spark- und Halo-Stilselektoren.	Standardwerte für Flex 3:
		`""`
		(In Flex 3 sind keine Namespace-Deklarationen in CSS-Dateien erforderlich.)
		Standardwerte für Flex 4:
		`@namespace s "library://ns.adobe.com/flex/spark"; @namespace mx "library://ns.adobe.com/flex/halo";`

Beispiele für Vorlagendateien

Im folgenden Beispiel werden eine MXML-Komponentendateivorlage und ein Listing für eine neue MXML-Komponentendatei gezeigt, die aus der Vorlage generiert wurden.

Beispiel einer Dateivorlage für eine MXML-Komponentendatei

```
${xml_tag}
<!--
* ADOBE SYSTEMS Confidential
*
* Copyright ${year}. All rights reserved.
*
* ${user}
* ${project_name}
* Created ${date}
*
-->
<${component} ${xmlns}${wizard_attributes}>
    ${wizard_tags}

    <${ns_prefix}Script>
    <![CDATA[

    ]]>
    </${ns_prefix}Script>
</${component}>
```

Neue MXML-Komponentendatei, die aus der Beispielvorlage generiert wurde

```
<?xml version="1.0" encoding="utf-8"?>
<!--
* ADOBE SYSTEMS Confidential
*
* Copyright 2009. All rights reserved.
*
* jdoe
* FileTemplates
* Created Jul 13, 2009
*
-->

<s:Group xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/halo" width="400" height="300">
    <s:layout>
        <s:BasicLayout/>
    </s:layout>

    <fx:Script>
        <![CDATA[

        ]]>
    </fx:Script>
</s:Group>
```

Aus der Verwendung generieren

Mit der Schnellhilfe können Sie Stubcode für eine undefinierte Methode, Variable oder Klasse in Ihrem Code generieren. Der generierte Stubcode kann als Platzhalter für den Code verwendet werden, den Sie später implementieren, ohne dass dieser Code inkompatibel wird. Wie Sie den von Flash Builder vordefinierten Stubcode anpassen, erfahren Sie unter „Codevorlagen“ auf Seite 57.

Sie rufen die Schnellhilfe über den Tastaturbefehl Strg+1 (Windows) bzw. Befehlstaste+1 (Mac OS) auf.

Erstellen Sie mit der Schnellhilfe Stubcode in der relevanten Klasse bzw. im relevanten MXML-Skriptblock, indem Sie eine der folgenden Aktionen wählen:

Methode generieren Erstellt eine Methode

Angenommen, der Code lautet wie folgt:

```
private function genFunc():void
{
bar();
}
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die `bar();` enthält, und drücken Sie Strg+1. Eine Option zum Erstellen von Funktionen wird angezeigt. Wenn Sie diese Option wählen, wird wie folgt eine neue Funktion erstellt:

```
private function bar():void{}
```

Sie können außerdem eine Funktion für eine undefinierte Funktion in einer referenzierten Klasse generieren. Angenommen, in der referenzierten Klasse „MyClass“ ist wie folgt die undefinierte Funktion „setValue()“ enthalten:

```
MyClass cls = new MyClass();
cls.setValue(5);
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die „setValue “enthält, und drücken Sie Strg+1. Eine Option zum Erstellen von Funktionen wird angezeigt. Wenn Sie diese Option wählen, wird wie folgt eine neue Funktion namens `setValue(int)` in der referenzierten MyClass erstellt:

```
 private function setValue(param0:int):void
{
    // TODO Auto Generated method stub
}
```

Variable generieren Erstellt eine Eigenschaft.

Angenommen, im Code ist wie folgt die undefinierte Variable `i` enthalten:

```
public function MyClass
{
        i;
}
```

Platzieren Sie den Cursor an einer beliebigen Stelle der Codezeile, die `i;` enthält, und drücken Sie Strg+1. Optionen zum Erstellen einer lokalen Variablen oder eines Felds werden angezeigt.

Wenn Sie die Option zum Erstellen einer lokalen Variablen wählen, wird die Variable wie folgt erstellt:

```
var i:Object;
```

Durch Wahl der Option zum Erstellen eines Felds wird wie folgt eine Variable auf Klassenebene erstellt:

```
private var i:Object;
```

Sie können außerdem eine Eigenschaft für eine undefinierte Variable in einer referenzierten Klasse generieren. Angenommen, in der referenzierten Klasse „MyClass“ ist wie folgt die undefinierte Variable „aValue“ enthalten:

```
MyClass cls = new MyClass();
cls.aValue = "str";
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die `aValue` enthält, drücken Sie Strg+1 und wählen Sie „Feldvariable erstellen“. Eine Eigenschaft namens `aValue` vom Typ String wird wie folgt in der referenzierten Klasse „MyClass“ erstellt:

```
private var aValue:String;
```

Klasse/Schnittstelle generieren Erstellt eine Klasse oder Schnittstelle.

Angenommen, im Code ist `Foo` undefiniert:

```
public function myFunction():Foo;
    {
    }
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die `Foo;` enthält, und drücken Sie Strg+1. Optionen zum Erstellen einer Klasse oder Schnittstelle mit dem Namen `Foo` werden angezeigt. Wählen Sie eine dieser Optionen, um entweder den Assistenten zum Erstellen einer neuen ActionScript-Klasse oder den Assistenten zum Erstellen einer neuen ActionScript-Schnittstelle zu öffnen. Geben Sie die nötigen Informationen ein und klicken Sie auf „Fertig stellen“. Nachdem Sie auf „Fertig stellen“ geklickt haben, wird eine Klasse oder Schnittstelle mit dem Namen `Foo` erstellt.

Wenn Sie eine neue Klasse generieren, können Sie eine ActionScript-Klasse mit einem parametrisierten Konstruktor erstellen.

Angenommen, Sie arbeiten mit Code wie dem folgenden:

```
Private function func(): void {
    New NewClass("str1");
}
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die „NewClass("str1")“ enthält, drücken Sie Strg+1 und wählen Sie „Klasse erstellen“. Eine ActionScript-Klasse mit einem parametrisierten Konstruktor wird erstellt. Wenn Sie jedoch eine Superklasse für die ActionScript-Klasse angeben, wird kein parametrisierter Konstruktor erstellt.

Adobe Community-Experte Paul Robertson schrieb einen Artikel über die Verwendung der Schnellhilfe für externe Klassen und Benutzeroberflächen.

Ereignisprozedur generieren Erstellt Ereignisprozedurfunktionen.

Angenommen, Sie arbeiten mit Code wie dem folgenden:

```
public function MyClass
    {
        Var button:Button = new Button();
        button.addEventListener(DragEvent.DRAG,dragHandlerFunction);
    }
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die `dragHandlerFunction` enthält, und drücken Sie Strg+1. Wählen Sie dann die Schnellhilfeoption zum Erstellen der Ereignisprozedur. Die Ereignisprozedurfunktion wird wie folgt erstellt:

```
protected function dragHandlerFunction (event:DragEvent):void
{
}
```

Importanweisung aus der Verwendung generieren Erstellt eine Importanweisung.

Im Code könnte z. B. der Variablentyp `Button` undefiniert sein:

```
<fx:Script>
    <![CDATA[
        var btn:Button;
    ]]>
</fx:Script>
```

Platzieren Sie den Cursor an einer beliebigen Stelle in der Codezeile, die `var btn:Button` enthält, und drücken Sie Strg+1. Sie erhalten eine Option zum Importieren von `Button`, sofern eine Klasse mit dem Namen `Button` im Projekt verfügbar ist. Die Importanweisung wird wie folgt erstellt:

```
import spark.components.Button;
```

Sie können Importanweisungen u. a. für Funktionsargumente und -rückgabetypen erstellen.

Get-/Set-Accessor-Funktionen generieren

Mit Get- und Set-Accessor-Funktionen (Getter und Setter) können Sie Klasseneigenschaften für die Klasse privat halten. Sie ermöglichen es Benutzern der Klasse, auf diese Eigenschaften zuzugreifen, als ob sie auf eine Klassenvariable zugreifen würden (anstatt eine Klassenmethode aufzurufen).

Flash Builder kann Get- und Set-Accessor-Funktionen in ActionScript für Klassenvariablen generieren. Sie können eine bindbare Eigenschaft auswählen und Get- und Set-Accessor-Funktionen für diese Eigenschaft generieren. Sie können beim Generieren des Codes auch einen benutzerdefinierten Ereignisnamen festlegen.

Anleitung zum Erstellen von Get- oder Set-Accessor-Funktionen

1. Platzieren Sie den Cursor in einer geöffneten ActionScript-Datei im Quelleditor auf eine Klassenvariable.
2. Wählen Sie entweder im Flash Builder-Menü oder im Kontextmenü „Quelle“ > „Getter/Setter generieren“.
3. Geben Sie im Dialogfeld „Getter/Setter“ Details für die Accessor-Funktionen an und klicken Sie auf „OK“.

   Hinweis: *Wenn Sie den generierten Code anzeigen möchten, klicken Sie auf „Vorschau“ und anschließend auf „OK“.*

Wie Sie den von Flash Builder generierten vordefinierten Code anpassen, erfahren Sie unter „Codevorlagen“ auf Seite 57.

Wenn Sie Getter und Setter generieren, bietet Flash Builder die folgenden Optionen:

- Klassenvariable privat machen.

  Klassenvariablen haben typischerweise privaten Zugriff.

- Klassenvariable umbenennen und einen Unterstrich am Anfang des Variablennamens vorschlagen.

  Standardmäßig beginnen private Klassenvariablen mit einem Unterstrich.

- Accessor-Funktionen umbenennen.
- Geben Sie eine bindbare Eigenschaft und einen benutzerdefinierten Ereignisnamen an.

  Wenn Sie eine bindbare Eigenschaft angeben, wird ein `[Bindable]`-Tag oberhalb der generierten Accessor-Funktion im generierten Code definiert.

- Festlegen, ob sowohl Get- als auch Set-Accessor-Funktionen generiert werden sollen.
- Geben Sie den Namespace-Wert für die Accessor-Funktion an.
- Position der Accessor-Funktion an einer der folgenden Positionen festlegen:
  - Vor der ersten Methode
  - Nach der letzten Methode
  - Vor Variablendeklarationen
- Zeigen Sie den generierten Code in einer Vorschau an.

Weitere Informationen zu Get- und Set-Accessor-Funktionen finden Sie unter Get- und Set-Accessor-Methoden in der *ActionScript 3.0-Reference für die Adobe Flash-Plattform*.

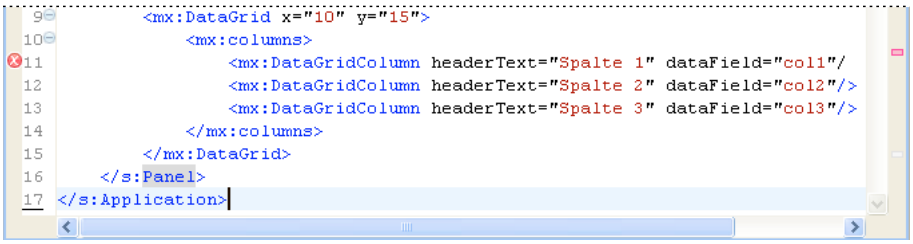
Get- oder Set-Accessor-Funktionen generieren

Auf Syntaxfehler prüfen

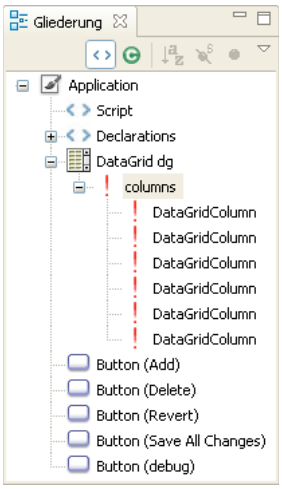
Beim Eingeben von Code erkennt und meldet der Compiler von Flash Builder syntaktische Fehler. Diese müssen vor dem Ausführen der Anwendung korrigiert werden. Voreinstellungen für Syntaxhervorhebung können leicht angepasst werden.

Wenn Codesyntaxfehler erkannt werden, werden Sie hierüber auf folgende Weisen informiert:

- Neben der Codezeile wird ein Fehlerindikator hinzugefügt, wie das folgende Beispiel zeigt:



- Die Gliederungsansicht weist wie im folgenden Beispiel gezeigt mit einem Ausrufezeichen in den betroffenen Codezeilen auf den Fehler hin:



- Die Problemansicht listet ein Fehlersymbol und eine Meldung auf. Wenn Sie auf die Fehlermeldung doppelklicken, wird die Codezeile wie im folgenden Beispiel gezeigt im Editor gesucht und hervorgehoben:

Kodierungssyntaxfehler werden beim Erstellen von Projekten gefunden. Wenn Sie Syntaxfehler vor dem Ausführen der Anwendung nicht beheben, werden Sie gewarnt, dass Fehler vorliegen. Je nach Art und Schweregrad der Fehler kann die Anwendung unter Umständen erst dann richtig ausgeführt werden, wenn die Fehler behoben wurden.

Voreinstellungen für Syntaxhervorhebung anwenden

❖ Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Editoren“ > „Syntaxhervorhebung“.

Die Standardschriftfarben können auch über die Voreinstellungen auf den Seiten „Texteditoren“ und „Farben und Schriftarten“ konfiguriert werden (siehe „Voreinstellungen > „Allgemein“ > „Darstellung“ > „Farben und Schriftarten“. Siehe auch „Voreinstellungen“ > „Allgemein“ > „Editoren“ > „Texteditoren“).

Unbekannte Verweise hervorheben

Während Sie ActionScript-Code eingeben, generiert Flash Builder automatisch Fehleranmerkungen für undefinierte IDs im Code. Fehleranmerkungen werden durch kenntlich gemacht.

Undefinierte Variablen und Methoden lassen sich dadurch schnell identifizieren, bevor Sie die Datei speichern und der Compiler den Fehler generiert.

Die Markierung unbekannter Verweisfehler erleichtert Ihnen außerdem das Auffinden von Stellen im Code, an denen Sie Stubcode für undefinierte Methoden, Variablen oder Klassen generieren können. Weitere Informationen finden Sie unter „Aus der Verwendung generieren“ auf Seite 72.

Die Markierung unbekannter Verweisfehler ist in Flash Builder standardmäßig aktiviert. Um sie zu deaktivieren, wählen Sie im Dialogfeld „Voreinstellungen“ „Flash Builder“ > „Editoren“ und deaktivieren Sie die Option „Fehler bei Eingabe melden“.

Verweise suchen und Code refaktorieren

Flash Builder enthält erweiterte Suchfunktionen, die von ihrer Funktionalität her über einen einfachen Such- und Ersetzungsvorgang hinausgehen. Damit Sie die Einsatzweise von Funktionen, Variablen oder anderen IDs leichter verstehen, können Sie in Flash Builder Verweise oder Deklarationen für IDs in ActionScript- und MXML-Dateien, Projekten oder Arbeitsbereichen suchen und markieren. Mit der Refaktorierung können Sie folgende Kennungen im Code umbenennen und alle Verweise darauf aktualisieren:

- Variablen
- Funktionen
- Typen (Schnittstelle, Klasse)
- Accessors (Getter/Setter)
- Attribute
- Metadaten in MXML (Effekte, Ereignisse, Stile)

Verweise markieren

1. Klicken Sie im Quellmodus in der Symbolleiste auf die Schaltfläche „Fundstellen markieren“.
2. Klicken Sie im Editor auf eine ID. Alle Instanzen werden entsprechend den Einstellungen unter „Voreinstellungen“ markiert.

Um das Aussehen von markierten Verweisen zu ändern, wählen Sie im Dialogfeld „Voreinstellungen“ die Kategorie „Allgemein“ > „Editoren“ > „Texteditoren“ > „Anmerkungen“. Weitere Informationen zu Markierungen finden Sie unter „Markierungen verwenden“ auf Seite 86.

Alle Verweise oder Deklarationen suchen

1. Klicken Sie im Quellmodus des Editors auf eine ID.
2. Wählen Sie im Hauptmenü „Suchen“ > „Verweise“ bzw. „Suchen“ > „Deklarationen“. Wählen Sie dann „Datei“, „Projekt“ oder „Arbeitsbereich“. Die Fundstellen werden in der Suchansicht angezeigt.

Code refaktorieren

1. Klicken Sie im Quellmodus des Editors auf eine ID.

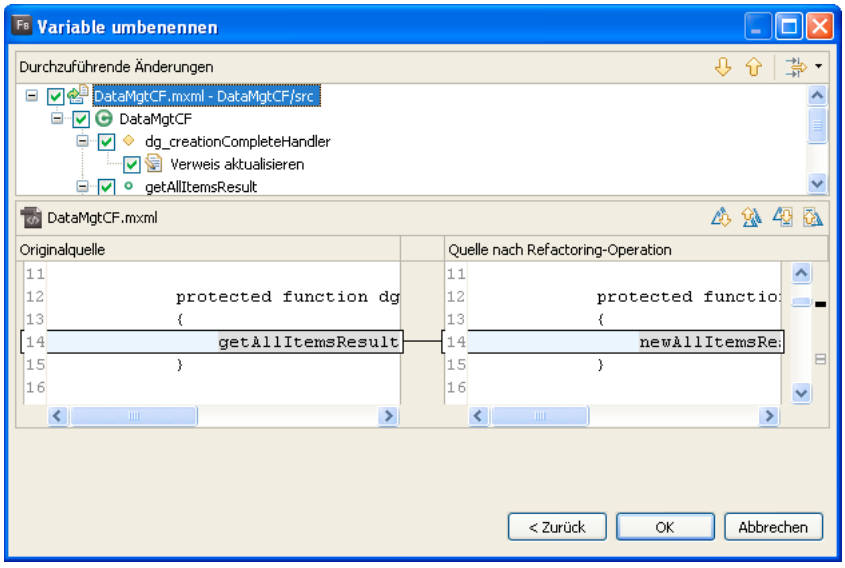
2 Wählen Sie im Hauptmenü „Quelle“ > „Refaktorieren“ > „Umbenennen“.

3 Geben Sie einen neuen Namen ein.

Flash Builder prüft die Umbenennungsvorbedingungen und fordert Sie auf, Probleme zu bestätigen, bevor der Umbenennungsvorgang erfolgt. Vorbedingungen umfassen Folgendes:

- Verweise können nicht in schreibgeschützten Dateien umbenannt werden;
- Alle Dateien müssen gespeichert sein;
- Wenn ein Projekt Erstellungsfehler hat, erscheint eine Warnung;
- Der neue Name muss innerhalb des Bereichs liegen, der durch den Typ des Elements und seine Position bestimmt ist; Name-Shadowing-Fehler werden ebenfalls angemerkt;
- Der neue Name muss eine gültige ID sein;
- Der in einer SWC-Datei definierte Verweis muss einen Quellanhang enthalten.

4 Um die Änderung zu sehen, klicken Sie auf „Vorschau“. Hierdurch wird der Original- und der refaktorierte Code angezeigt. Um mit der Änderung des Codes fortzufahren, klicken Sie auf „OK“.



In CSS-Dateien refaktorieren

Wenn CSS-Dateien Verweise auf ActionScript- oder MXML-Dateien enthalten und Sie die ActionScript- oder MXML-Dateien an einen anderen Speicherort verschieben, aktualisiert Flash Builder automatisch die CSS-Dateien mit Verweisen auf die neuen Namen oder Speicherorte. Danach können Sie die Änderungen im Dialogfeld „Vorschau“ überprüfen. Wenn Sie hier auf „OK“ klicken, können Sie mit den Änderungen in Ihrem Code arbeiten.

Formatieren, navigieren und Code organisieren

Die Flash Builder-Editoren bieten viele Tastaturbefehle zum Navigieren durch den Code. Sie können beispielsweise Codeblöcke ein- und ausklappen, die Quelle von Codedefinitionen öffnen und nach Typen suchen und diese öffnen. Codenavigation bietet die Möglichkeit, ein Codeelement (z. B. einen Verweis auf eine benutzerdefinierte Komponente in einer MXML-Anwendungsdatei) auszuwählen und zur Quelle der Codedefinition zu gehen, egal wo sich diese im Projekt, Arbeitsbereich oder Pfad befindet.

Mehrzeilige Codeblöcke können ein- und ausgeblendet werden, damit Sie Dokumente mit komplexem Code leichter lesen, verwalten und einfacher darin navigieren können. In Flash Builder wird das Ein- und Ausblenden mehrzeiliger Codeanweisungen mit *Codeeinklappung* und *Codeausklappung* bezeichnet.

Code formatieren, einrücken und kommentieren

Wenn Sie Code schreiben, rückt Flash Builder Codezeilen automatisch ein, um die Lesbarkeit zu verbessern, hebt Codeelemente farblich hervor und stellt viele Befehle zur schnellen Formatierung von Code während der Eingabe (z. B. zum Hinzufügen eines Blockkommentars) zur Verfügung.

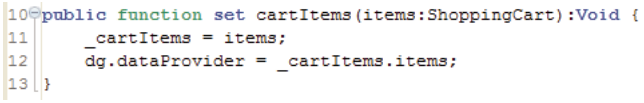
Zum Ändern der Standardformatierung wählen Sie im Dialogfenster„Voreinstellungen“ die Option „Flash Builder“ > „MXML-Code“ > „Formatierung“. Reihenfolge und Gruppierung der Attribute lassen sich ändern.

Wenn Sie MXML- oder ActionScript-Code in den Codeeditor einfügen, rückt Flash Builder den Code automatisch entsprechend den von Ihnen ausgewählten Voreinstellungen ein. Sie können auch angeben, ob bzw. wie ausgewählte Codeblöcke eingerückt werden sollen.

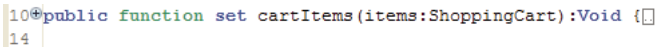
Um die Einrückung anzupassen, wählen Sie im Dialogfenster „Voreinstellungen“ die Option „Flash Builder“ > „Editoren“. Sie können Art und Größe der Einrückung bestimmen.

Codeblöcke setzen, aus- und einklappen

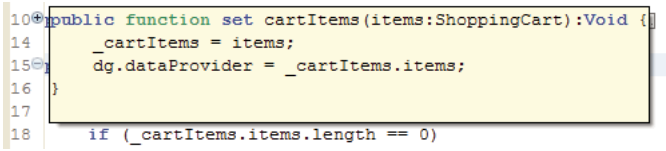
1 Klicken Sie am linken Rand des Editors auf das Einklapp- (-) bzw. das Ausklappsymbol (+).



Wenn Sie einen Codeblock einklappen, wird bis auf die erste Codezeile alles ausgeblendet.



Um den Codeblock wieder sichtbar zu machen, müssen Sie ihn ausklappen. Zeigen Sie mit der Maus auf das Ausklappsymbol (+), um den gesamten Codeblock in einer QuickInfo anzuzeigen.



2 Standardmäßig ist in Flash Builder die Codeeinklappung aktiviert. Um die Codeeinklappung zu deaktivieren, öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Editoren“. Deaktivieren Sie dann die Option „Codeeinklappung aktivieren“.

Codeblöcke einrücken

Der Editor formatiert Codezeilen während der Eingabe, um die Lesbarkeit des Codes zu verbessern und ein effizientes Schreiben von Code zu ermöglichen. Sie können auch die Tabulatortaste verwenden, um einzelne Codezeilen manuell einzurücken.

Wenn Sie Codeblöcke in Flash Builder kopieren und einfügen, rückt Flash Builder den Code automatisch entsprechend den von Ihnen ausgewählten Voreinstellungen ein.

Wenn Sie einen Codeblock in einem Arbeitsschritt einrücken möchten, können Sie hierzu die Editorbefehle „Einrücken“ und „Ausrücken“ verwenden.

Codeblock aus- oder einrücken

1. Wählen Sie im Editor einen Codeblock aus.
2. Wählen Sie „Quelle“ > „Einrücken“ bzw. „Quelle“ > „Ausrücken“.
3. Drücken Sie die Tabulatortaste bzw. die Umschalt- und die Tabulatortaste, um Codeblöcke ein- bzw. auszurücken.

Einrückvoreinstellungen festlegen

1. Öffnen Sie das Dialogfenster „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Einrückung“.
2. Wählen Sie den Einrücktyp (Tabulatoren oder Leerzeichen) aus und geben Sie die Einrückungsgröße und die Tabulatorgröße an.

Kommentare und Kommentarblöcke hinzufügen

Mithilfe von Optionen im Menü „Quelle“ oder Tastenkombination können Sie Kommentare hinzufügen oder entfernen. Sie können die folgenden Arten von Kommentaren hinzufügen:

- Quellkommentare für ActionScript (`//`)
- Blockkommentare für ActionScript (`/* */`)
- ASDoc-Kommentare für ActionScript (`/** */`)
- Blockkommentare für MXML (`<!---->`)
- CDATA-Block für MXML (`<![CDATA[ ]]>`)

Kommentare können in ActionScript-Code ein- und ausgeblendet werden.

Adobe Community-Experte Paul Robertson schrieb einen Artikel über die Verwendung des Blockauswahlmodus.

Kommentare in ActionScript-Code ein- und ausblenden

1. Wählen Sie im Editor eine oder mehrere Zeilen des ActionScript-Codes aus.
2. Drücken Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) sowie Umschalttaste+C, um Kommentare im C-Stil hinzuzufügen oder zu entfernen.
3. Drücken Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) sowie /, um Kommentare im C++-Stil hinzuzufügen oder zu entfernen.

XML-Kommentare in MXML-Code hinzufügen

1. Wählen Sie im Editor eine oder mehrere Zeilen des MXML-Codes aus.
2. Drücken Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) sowie Umschalttaste+C, um einen Kommentar hinzuzufügen.

CDATA-Blöcke in MXML-Code hinzufügen

1. Wählen Sie im Editor eine oder mehrere Zeilen des MXML-Codes aus.
2. Drücken Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) sowie Umschalttaste+D, um einen Kommentar hinzuzufügen.

In Code navigieren und ihn inspizieren

Anwendungen jeglicher Komplexität werden stets viele Ressourcen und Codezeilen enthalten. Flash Builder bietet eine Reihe von Funktionen zum Navigieren im Code und zum Inspizieren seiner Elemente.

Codedefinitionen öffnen

In Flash Builder können Sie die Quelle einer externen Codedefinition von der Stelle in Ihrem Code aus öffnen, von der auf die externe Definition verwiesen wird. Wenn Sie beispielsweise eine benutzerdefinierte MXML-Komponente erstellen und in die MXML-Anwendung importieren, können Sie den Verweis auf die MXML-Komponente auswählen und die Quelldatei im Editor öffnen.

Quelle einer Codedefinition öffnen

1. Wählen Sie den Codeverweis im Editor aus.
2. Wählen Sie im Menü „Navigieren“ den Befehl „Zu Definition gehen“.

   Sie können den Tastenbefehl F3 verwenden.

   Die Quelldatei, die die Codedefinition enthält, wird im Editor geöffnet.

Flash Builder unterstützt auch Hyperlink-Codenavigation.

Quelle einer Codedefinition durch Hyperlinknavigation öffnen

1. Wählen Sie den Codeverweis im Editor aus.
2. Drücken Sie die Strg-Taste (Windows) bzw. die Befehlstaste (Mac OS) und zeigen Sie bei gedrückter Strg-/Befehlstaste mit der Maus auf den Codeverweis, um den Hyperlink anzuzeigen.
3. Um zum Codeverweis zu navigieren, klicken Sie auf den Hyperlink.

Gliederungsansicht verwenden

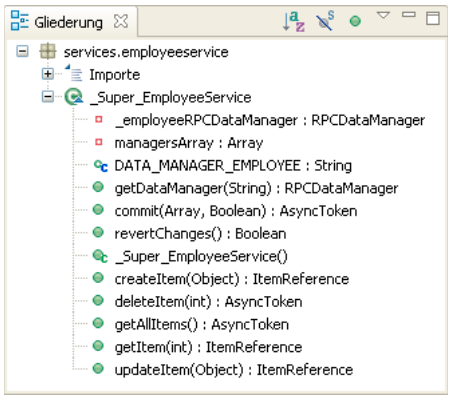
Die Gliederungsansicht ist Teil der Flash-Entwicklungsperspektive (siehe „Flash-Entwicklungsperspektive“ auf Seite 8) und daher verfügbar, wenn Sie Code bearbeiten und die Anwendung entwickeln. In der Gliederungsansicht können Sie die Struktur von MXML-, ActionScript- und CSS-Dokumenten leichter inspizieren und durch sie navigieren.

Die Gliederungsansicht enthält drei Modi: Klasse, MXML und CSS. Im Klassenmodus zeigt die Gliederungsansicht die Struktur des Codes (Klassen, Member-Variablen, Funktionen usw.) an. Im MXML-Modus zeigt die Gliederungsansicht die MXML-Struktur (Tags, Komponenten, Steuerelemente usw.) an. Im CSS-Modus werden CSS-Selektoren und darin verschachtelte Eigenschaften angezeigt.

Wenn Sie ein Element in der Gliederungsansicht auswählen, wird es im Editor gesucht und hervorgehoben. Hierdurch können Sie viel leichter durch Code navigieren.

Gliederungsansicht im Klassenmodus

Wenn Sie ein ActionScript-Dokument (oder in einem MXML-Dokument enthaltenes ActionScript) bearbeiten, zeigt die Gliederungsansicht die Struktur des Codes an. Hierzu gehören Importanweisungen, Pakete, Klassen, Schnittstellen, nicht in Funktionen enthaltene Variablen und Funktionen. Diese Ansicht zeigt keine Metadaten, Kommentare, Namespace-Deklarationen und auch nicht den Inhalt von Funktionen an.



In der Gliederungsansicht stellen Knoten und Objekte in der Ordnerstruktur sowohl die verschiedenen Sprachelementtypen als auch ihre Sichtbarkeit dar. Rote Symbole weisen beispielsweise auf private Elemente hin, grüne auf öffentliche Elemente und gelbe Symbole bedeuten, dass das Element weder privat noch öffentlich ist.

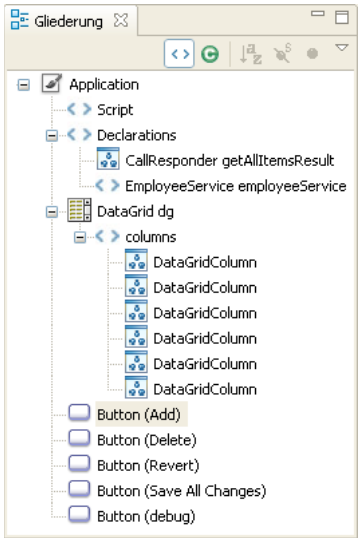
Symbolleiste der Gliederungsansicht im Klassenmodus

Im Klassenmodus enthält die Symbolleiste der Gliederungsansicht Befehle zum Sortieren und Filtern, wie das folgende Beispiel zeigt:

Gliederungsansicht im MXML-Modus

Wenn Sie ein MXML-Dokument bearbeiten, das sowohl MXML- als auch ActionScript-Code enthalten kann, sind in der Gliederungsansicht der Klassen- und der MXML-Modus verfügbar.

Im MXML-Modus stellt jedes Element in der Gliederungsansicht ein MXML-Tag dar. Die folgenden Tag-Typen werden angezeigt: Komponenten, Steuerelemente, nicht-visuelle Tags wie z. B `WebService` oder `State`, Komponenteneigenschaften, die als untergeordnete Tags (z. B. Layoutbeschränkungen) ausgedrückt sind, und Compiler-Tags wie z. B. `Model`, `Array` und `Script`.



Die Gliederungsansicht im MXML-Modus zeigt keine Kommentare, CSS-Regeln und Eigenschaften und auch keine als Attribute ausgedrückte Komponenteneigenschaften (im Gegensatz zu untergeordneten Tags, die angezeigt werden), an.

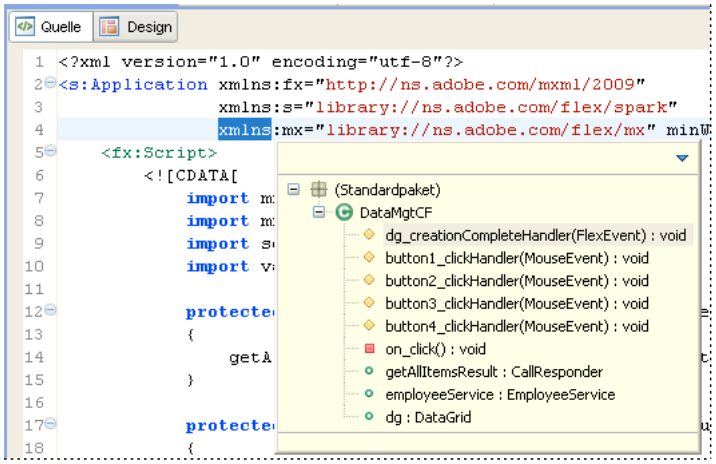
Symbolleiste der Gliederungsansicht im MXML-Modus

Das Symbolleistenmenü der Gliederungsansicht im MXML-Modus enthält zusätzliche Befehle, mit denen Sie zwischen der MXML- und der Klassenansicht wechseln können.

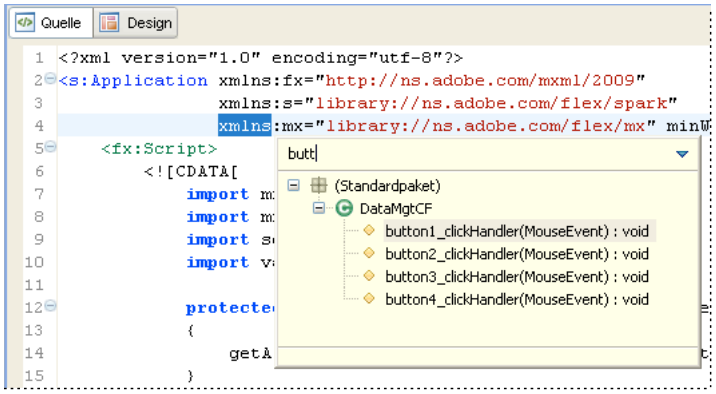
Wählen Sie zum Wechseln zwischen den beiden Ansichten in der Symbolleiste die Option „MXML-Ansicht anzeigen“ oder „Klassenansicht anzeigen“.

Schnellgliederungsansicht im Editor

Vom ActionScript- und MXML-Editor aus können Sie die Schnellgliederungsansicht aufrufen, die die Gliederungsansicht im Klassenmodus anzeigt. Die Schnellgliederungsansicht wird in einem Popupfenster im Editor (nicht in einer separaten Ansicht) angezeigt und über sie können Sie schnell zu Code navigieren und diesen inspizieren.



Die Schnellgliederungsansicht hat den gleichen Inhalt wie der Klassenmodus, enthält aber auch einen Texteingabebereich, über den Sie die angezeigten Elemente filtern können. Wenn Sie beispielsweise einen Elementnamen in die Schnellgliederungsansicht eingeben, werden nur die Elemente angezeigt, die diese Zeichen enthalten.



Die Schnellgliederungsansicht enthält keine Befehle zum alphabetischen Sortieren oder Ausblenden von Elementen.

Genau wie in der Gliederungsansicht wird das von Ihnen ausgewählte Element im Editor gesucht und dort hervorgehoben.

Schnellgliederungsansicht öffnen

- Öffnen Sie im Editor ein ActionScript- oder MXML-Dokument und wählen Sie im Menü „Navigieren“ den Befehl „Schnellgliederung“.

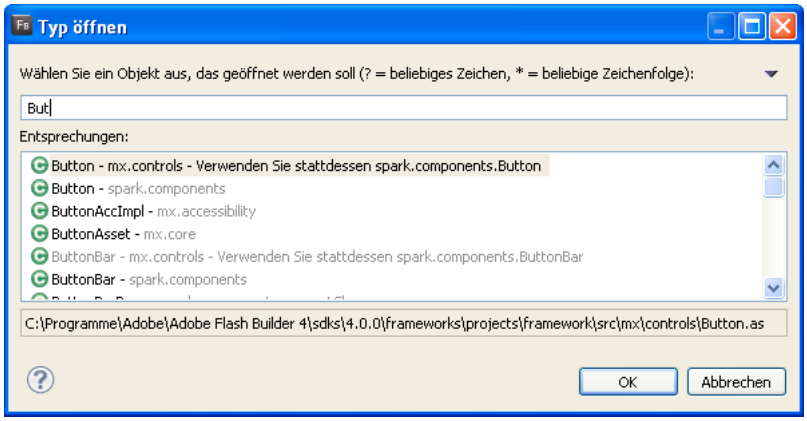
  Sie können dazu auch die Tastenkombination Strg/Befehl+O verwenden.

Schnellgliederungsansicht schließen

❖ Die Schnellgliederungsansicht wird geschlossen, wenn Sie außerhalb dieser Ansicht navigieren. Sie können auch die Esc-Taste drücken, um die Schnellgliederungsansicht zu schließen.

Klassen durchsuchen und anzeigen

Mit dem Dialogfeld „Typ öffnen“ können Sie alle verfügbaren Klassen (inklusive Flex-Frameworkklassen) in Ihrem Projekt durchsuchen. Wählen Sie im Dialogfeld „Typ öffnen“ eine Klasse aus, um die Implementierung anzuzeigen.


Typ öffnen
Wählen Sie ein Objekt aus, das geöffnet werden soll (? = beliebiges Zeichen, * = beliebige Zeichenfolge):
But
Entsprechungen:
Button - mx.controls - Verwenden Sie stattdessen spark.components.Button
Button - spark.components
ButtonAccImpl - mx.accessibility
ButtonAsset - mx.core
ButtonBar - mx.controls - Verwenden Sie stattdessen spark.components.ButtonBar
ButtonBar - spark.components
C:\Programme\Adobe\Adobe Flash Builder 4\sdks\4.0.0\frameworks\projects\framework\src\mx\controls\Button.as
OK
Abbrechen

Dialogfeld „Typ öffnen“

Mithilfe des Dialogfelds „Typ öffnen“ können Sie auch eine Klasse als Basisklasse für eine neue ActionScript-Klasse oder eine neue MXML-Komponente wählen.

Im Dialogfenster „Typ öffnen“ können Sie Klassen nach bestimmtem Text filtern und dabei auch Platzhalter angeben. Im Dialogfeld wird eine Farbkodierung zur Kennzeichnung der empfohlenen Typen und der ausgeschlossenen Typen verwendet. Empfohlene Typen werden in Grau angezeigt. Ausgeschlossene Typen werden braun dargestellt.

Empfohlene Typen sind Typen, deren Klassen im Standard-Namespace für ein Projekt zur Verfügung stehen. Beispielsweise werden unter bestimmten Bedingungen ausschließlich Spark-Komponenten zugelassen. In anderem Kontext werden sowohl Spark- als auch Halo-Komponenten zugelassen.

Ausgeschlossene Typen sind diejenigen Klassen, die im Standard-Namespace nicht für ein Projekt zur Verfügung stehen.

Dialogfeld „Typ öffnen“ öffnen

- (Klassen durchsuchen) So durchsuchen Sie Klassen und zeigen ihre Implementierung an:
  1. Wählen Sie im Flash Builder-Menü „Navigieren“ > „Typ öffnen“.
  2. (Optional) Geben Sie Text ein oder wählen Sie Filter, um die in der Liste angezeigten Klassen zu modifizieren.
  3. Wählen Sie eine Klasse aus, um den Quellcode anzuzeigen.

     Sie können den Quellcode von Klassen nicht im Flex-Framework modifizieren.
- (Neue ActionScript-Klassen) Gehen Sie folgendermaßen vor, wenn Sie eine Basisklasse für eine neue ActionScript-Klasse auswählen:
  1. Wählen Sie „Datei“ > „Neu“ > „ActionScript-Klasse“.
  2. Klicken Sie neben dem Textfeld für „Superklasse“ auf „Durchsuchen“.
  3. (Optional) Geben Sie Text ein oder wählen Sie Filter, um die in der Liste angezeigten Klassen zu modifizieren.

4 Wählen Sie eine Basisklasse aus der Liste.

- (Neue MXML-Komponenten) Gehen Sie folgendermaßen vor, wenn Sie eine Basisklasse für eine neue MXML-Komponente auswählen:

1 Wählen Sie „Datei“ > „Neu“ > „MXML-Komponente“.

2 Wählen Sie in Ihrem Arbeitsbereich in der Liste der Projekte ein Projekt für eine neue MXML-Komponente aus und geben Sie einen Dateinamen an.

Welche Basiskomponenten verfügbar sind, hängt von den für ein Projekt konfigurierten Namespaces ab.

3 Klicken Sie neben dem Textfeld für „Basiert auf“ auf „Durchsuchen“.

Hinweis: *Löschen oder ändern Sie die Standard-Basisklasse, die im Textfeld „Basiert auf“ angezeigt wird, um die Auswahl zu erweitern.*

4 (Optional) Geben Sie Text ein oder wählen Sie Filter, um die in der Liste angezeigten Klassen zu modifizieren.

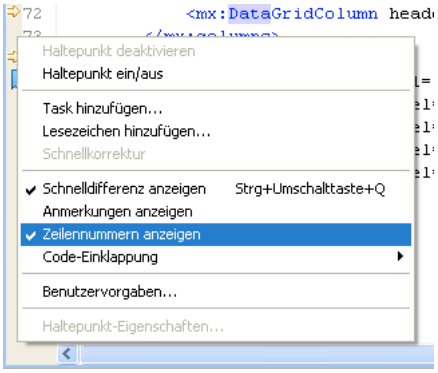
5 Wählen Sie eine Basiskomponente aus der Liste.

Zeilennummern einblenden

Sie können im Editor Zeilennummern hinzufügen, damit der Code leichter lesbar und navigierbar wird.

❖ Wählen Sie im Kontextmenü am Rand des Editors den Befehl „Zeilennummer anzeigen“.

Der Rand des Editors befindet sich zwischen der Markierungsleiste und dem Editor.



Markierungen verwenden

Markierungen sind Schnellzugriffe auf Codezeilen in einem Dokument, auf ein Dokument selbst oder auf einen Ordner. Markierungen geben Tasks, Lesezeichen und Fehler an und werden angezeigt und verwaltet. Wenn Sie eine Markierung auswählen, wird das verknüpfte Dokument im Editor geöffnet und optional kann die jeweilige Codezeile hervorgehoben werden.

In Flash Builder müssen Sie eine Datei speichern, um Fehlermarkierungen zu aktualisieren. Es werden nur Dateien geprüft, auf die von der Anwendung verwiesen wird. Die Syntax in einer isolierten Klasse, die an keiner Stelle im Code verwendet wird, wird nicht geprüft.

Die Workbench generiert die folgenden Task- und Fehlermarkierungen automatisch. Sie können Tasks und Lesezeichen manuell hinzufügen.

Tasks Taskmarkierungen sind ein Arbeitselement. Arbeitselemente werden automatisch von der Workbench generiert. Sie können einen Task manuell zu einer bestimmten Codezeile in einem Dokument oder zum Dokument selbst hinzufügen. Beispielsweise können Sie einen Task mit dem Namen „Skineigenschaften definieren“ erstellen, um

sich selbst daran zu erinnern, eine Komponenteneigenschaft zu definieren. Sie können auch allgemeine Tasks hinzufügen, die sich nicht direkt auf Ressourcen beziehen (z. B. „Benutzerdefinierte Komponente für Anmeldedialog für Mitarbeiter erstellen“). Die Taskansicht wird zum Verwalten aller Taskmarkierungen verwendet. Weitere Informationen hierzu finden Sie unter „Tasks hinzufügen“ auf Seite 87.

Fehler Fehlermarkierungen werden vom Compiler generiert und weisen auf verschiedenste ungültige Status hin. Beispielsweise werden vom Compiler generierte Syntaxfehler und Warnungen als Fehlermarkierungen in der Problemansicht angezeigt. Weitere Informationen finden Sie unter „Problemansicht“ auf Seite 11.

Lesezeichen Sie können einer Codezeile oder einer Ressource (einem Ordner oder einem Dokument) Lesezeichen manuell hinzufügen. Lesezeichen sind praktisch, um einen Überblick über Projekte zu behalten und schnell zu bestimmten Elementen in Projekten navigieren zu können. Die Lesezeichenansicht wird zum Verwalten aller Lesezeichen verwendet. Weitere Informationen finden Sie unter „Lesezeichen hinzufügen und löschen“ auf Seite 88.

Hinweis: *Die Task- und Lesezeichenansicht werden nicht standardmäßig in der Flash-Entwicklungsperspektive angezeigt. Weitere Informationen zum Hinzufügen dieser Ansichten finden Sie unter „Mit Ansichten arbeiten“ auf Seite 20.*

Markierungen navigieren

Markierungen sind Beschreibungen sowie Links zu Elementen in Projektressourcen. Markierungen werden vom Compiler automatisch erzeugt, um auf Probleme im Code hinzuweisen. Sie können auch manuell eingefügt werden, damit Sie bei Ihren Tasks und Codefragmenten den Überblick behalten. Markierungen werden in den jeweils zugehörigen Ansichten angezeigt und verwaltet. Sie können Markierungen im Projekt leicht über die Lesezeichen-, Problem- und Taskansicht finden und von dort zur Stelle gelangen, an der die Markierung gesetzt wurde.

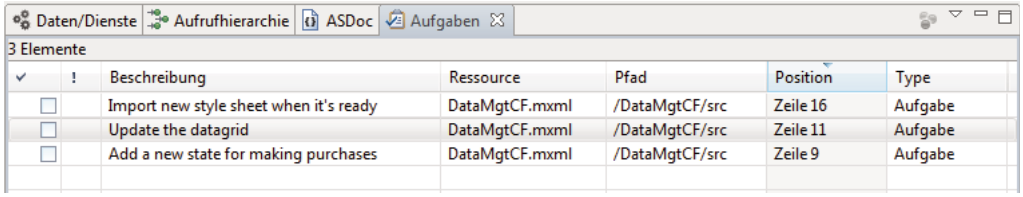
Position einer Markierung aufrufen

❖ Wählen Sie in der Lesezeichen-, Problem- oder Taskansicht eine Markierung.

Die Datei, die die Markierung enthält, wird gesucht und im Editor geöffnet. Wenn eine Markierung auf einer Codezeile gesetzt wurde, wird die entsprechende Zeile hervorgehoben.

Tasks hinzufügen

Tasks sind automatisch oder manuell generierte Arbeitsbereichelemente. Alle Aufgaben werden in der Taskansicht („Fenster“ > „Andere Ansichten“ > „Allgemein“ > „Tasks“) angezeigt und verwaltet, wie das folgende Beispiel zeigt:


Daten/Dienste
Aufrufhierarchie
ASDoc
Aufgaben
3 Elemente
! Beschreibung Ressource Pfad Position Type
Import new style sheet when it's ready DataMgtCF.mxml /DataMgtCF/src Zeile 16 Aufgabe
Update the datagrid DataMgtCF.mxml /DataMgtCF/src Zeile 11 Aufgabe
Add a new state for making purchases DataMgtCF.mxml /DataMgtCF/src Zeile 9 Aufgabe

Task einer Codezeile oder einer Ressource hinzufügen

1 Öffnen Sie eine Datei im Editor und suchen und markieren Sie dann die Codezeile, in der Sie einen Task hinzufügen möchten. Alternativ können Sie im Flex Paket-Explorer eine Ressource auswählen.

2 Klicken Sie in der Taskansicht in der Symbolleiste auf die Schaltfläche „Task hinzufügen“.

3 Geben Sie einen Namen für den Task ein, wählen Sie eine Priorität („Hoch“, „Normal“, „Niedrig“) aus und klicken Sie auf „OK“.

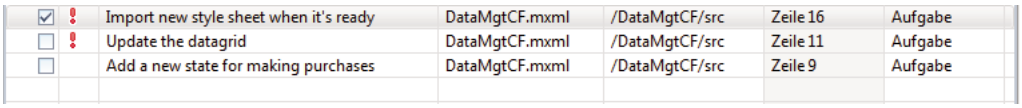
Hinweis: *Die Ressource im Flex Paket-Explorer zeigt nicht an, dass sie markiert wurde. Sie können alle Taskmarkierungen in der Taskansicht anzeigen und verwalten.*

Tasks abschließen und löschen

Wenn ein Task abgeschlossen ist, können Sie ihn entsprechend markieren und dann optional aus der Taskansicht löschen.

Task als abgeschlossen markieren

❖ Wählen Sie den gewünschten Task wie im folgenden Beispiel gezeigt in der Auswahlspalte der Taskansicht aus:



☑	!	Import new style sheet when it's ready	DataMgtCF.mxml	/DataMgtCF/src	Zeile 16	Aufgabe
☐	!	Update the datagrid	DataMgtCF.mxml	/DataMgtCF/src	Zeile 11	Aufgabe
☐		Add a new state for making purchases	DataMgtCF.mxml	/DataMgtCF/src	Zeile 9	Aufgabe

Task löschen

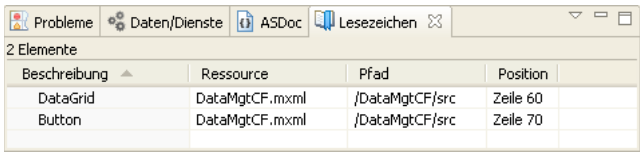
❖ Öffnen Sie in der Tasks-Ansicht das Kontextmenü des Tasks und wählen Sie „Löschen“.

Alle abgeschlossenen Tasks löschen

❖ Öffnen Sie in der Tasks-Ansicht das Kontextmenü und wählen Sie „Abgeschlossene Tasks löschen“.

Lesezeichen hinzufügen und löschen

Lesezeichen können verwendet werden, um einen Überblick über Projekte zu erhalten und schnell zu bestimmten Elementen in Projekten navigieren zu können. Alle Lesezeichen werden in der Lesezeichenansicht („Fenster“ > „Andere Ansichten“ > „Allgemein“ > „Lesezeichen“) angezeigt und verwaltet, wie das folgende Beispiel zeigt:



Lesezeichen einer Codezeile oder einer Ressource hinzufügen

1 Öffnen Sie eine Datei im Editor und suchen und markieren Sie dann die Codezeile, der Sie ein Lesezeichen hinzufügen möchten.

2 Wählen Sie im Hauptmenü „Bearbeiten“ > „Lesezeichen hinzufügen“.

3 Geben Sie einen Namen für das Lesezeichen ein und klicken Sie auf „OK“.

Neben der Codezeile wird ein Lesezeichensymbol ( ) hinzugefügt.

Hinweis: *Die Ressource im Flex Paket-Explorer zeigt nicht an, dass sie markiert wurde. Sie können alle Lesezeichen in der Lesezeichenansicht anzeigen und verwalten.*

Lesezeichen löschen

1 Wählen Sie in der Lesezeichenansicht das Lesezeichen aus, das gelöscht werden soll.

2 Klicken Sie mit der rechten Maustaste (Windows) bzw. bei gedrückter Ctrl-Taste (Mac OS) auf das Lesezeichen und wählen Sie „Löschen“.

Importanweisungen organisieren

Wenn Sie die Inhaltshilfe im MXML- und ActionScript-Editor verwenden, werden die Pakete, in denen sich Klassen befinden, automatisch in das Dokument importiert. Sie werden in der Reihenfolge hinzugefügt, in der sie in Code eingegeben wurden. Nicht verwendete bzw. nicht benötigte Importe werden automatisch entfernt.

Um den Code in ActionScript-Dokumenten übersichtlicher zu gestalten, können Sie Importanweisungen alphabetisch sortieren. Öffnen Sie dazu das Dialogfeld „Voreinstellungen“, wählen Sie „Flash Builder“ > „Editoren“ > „ActionScript-Code“ aus und aktivieren Sie dann die Option „Importe weiterhin verwalten“.

Importanweisungen sortieren

❖ Öffnen Sie im Editor ein ActionScript-Dokument, das Importanweisungen enthält, und wählen Sie im Menü „Quelle“ den Befehl „Importe verwalten“.

Sie können dazu auch die Tastenkombination Strg+Umschalttaste+O (Windows) oder Befehlstaste+Umschalttaste+O (Mac OS) verwenden.

Kapitel 4: Projekte in Flash Builder

Mit Adobe® Flash® Builder™ können Sie Projekte zum Erstellen von Web-, Desktop- und Mobilanwendungen anlegen, verwalten, verpacken und verteilen. Wenn Sie SWC-Dateien (SWC = Shared Component Library) generieren, können Sie Komponenten und andere Ressourcen in verschiedenen Anwendungen nutzen bzw. mit anderen Entwicklern austauschen. Sie können in Flash Builder auch direkt mit verschiedenen Versionen des Adobe Flex SDKs arbeiten.

Projekte in Flash Builder erstellen

Mit Flash Builder können Sie verschiedene Projekttypen erstellen, z. B. Flex, Flex auf Mobilgeräten, ActionScript und AIR-Projekte sowie andere Projekttypen. Flash Builder bietet einen Projekterstellungsassistenten, der Sie in den verschiedenen Schritten zur Eingabe des zu erstellenden Projekts, des Projektnamens, des Anwendungstyps (Web oder Desktop), der SDK-Versionen und anderen Optionen auffordert.

Informationen zum Erstellen eines ActionScript-Projekts finden Sie unter „ActionScript-Projekte“ auf Seite 93.

Informationen zum Erstellen von Bibliotheksprojekten finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 234.

Informationen zum Erstellen von Flash Professional-Projekten finden Sie unter „Verwenden von Flash Builder mit Flash Professional“ auf Seite 251.

Flex-Projekte

Mithilfe von Flex-Projekten können Sie Webanwendungen (wird in Flash Player ausgeführt) oder Desktopanwendungen (wird in Adobe AIR ausgeführt) erstellen. Es gibt Optionen zum Erstellen von MX- oder Flex-Projekten oder nur Spark-Projekten, die nur Spark-Komponenten verwenden.

Flex-Projekte erstellen

Führen Sie folgende Schritte zum Erstellen von grundlegenden Web- oder Desktopanwendungen durch.

1. Wählen Sie „Datei“ > „Neu“ > „Flex-Projekt“.
2. Geben Sie einen Projektnamen und Projektpfad ein.

   Der Standardspeicherort ist der aktuelle Arbeitsbereich.
3. Wählen Sie als Anwendungstyp entweder Web oder Desktop.
4. Verwenden Sie das standardmäßige Flex-SDK oder navigieren Sie zu einem anderen installierten SDK. Klicken Sie auf „Weiter“.
5. (Optional) Geben Sie die Servereinstellungen an.

   Siehe „Auf Datendienste zugreifen“ auf Seite 221.
6. Geben Sie einen Ausgabeordner an.

   Wenn Sie keinen Anwendungsserver angeben, befindet sich der Speicherort im Projektordner.

   Wenn Sie einen Anwendungsserver angeben, befindet sich der Ausgabeserver außerhalb des Projektordners. Normalerweise wird der Ausgabeordner bei den Dienstdateien abgelegt.
7. Klicken Sie auf „Fertig stellen“ oder „Weiter“, um zusätzliche Konfigurationsoptionen anzugeben.

8 (Optional) Geben Sie Erstellungspfade und andere Konfigurationsoptionen ein.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

9 Klicken Sie auf „Fertig stellen“.

Spielen Sie das Video zum Erstellen Ihrer ersten Desktop-Anwendung mit Flash Builder von Adobe-Guru James Ward ab.

Verwandte Themen

„Projekte in Flash Builder“ auf Seite 35

Flex-Projekte erstellen, die ausschließlich MX-Komponenten verwenden

Die Option „Nur MX“ ist nützlich, um Anwendungen zu erstellen, deren Design demjenigen von Anwendungen ähnelt, die mit der vorherigen Flex-Version (d. h. Flex 3) erstellt wurden, die aber dennoch Zugriff auf die neuesten Flex- und Flash Builder-Funktionen haben (wie z. B. die Syntax für Status, erweiterte CSS- und verbesserte Compiler-Funktionen sowie sonstige Sprachfunktionen).

Wenn Sie für ein Projekt „Nur MX“ angeben, sind Spark-Komponenten nicht für Anwendungen im Projekt verfügbar.

Sie können ein Flex -Projekt in ein „Nur MX“-Projekt konvertieren. Flash Builder schreibt jedoch keinen Code im Projekt um. Aktualisieren Sie Ihren Code manuell, um gegebenenfalls vorhandene Verweise auf Spark-Komponenten zu entfernen.

Reine MX-Flex-Projekte erstellen

1 Wählen Sie „Datei“ > „Neu“ > „Flex-Projekt“.

2 Geben Sie den Projektpfad und die Servereinstellungen wie unter „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96 beschrieben an.

3 Geben Sie auf der Seite „Erstellungspfad“ des Assistenten „Neues Flex-Projekt“ „Nur MX“ an.

4 Geben Sie weitere Erstellungspfadeigenschaften wie unter „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96 beschrieben an. Klicken Sie auf „Fertig stellen“.

Flex-Projekte in ein reine MX-Flex-Projekte umwandeln

1 Legen Sie das Projekt als aktives Projekt in Flash Builder fest.

Normalerweise öffnen Sie eine Quelldatei im Projekt, um das Projekt aktiv zu machen.

2 Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.

3 Geben Sie für „Komponentensatz“ die Option „Nur MX“ an. Klicken Sie auf „OK“.

4 Ändern Sie im Projekt den Anwendungscode, der auf Spark-Komponenten zugreift.

In einem „Nur MX“-Projekt können Sie nicht auf Spark-Komponenten verweisen.

Flex-Projekte erstellen, die ausschließlich Spark-Komponenten verwenden

Die Option „Nur Spark“ ist nützlich, um Anwendungen zu erstellen, die Flex- und Flash Builder-Funktionen wie die neue Syntax für Status, erweiterte CSS- und verbesserte Compiler-Funktionen sowie sonstige Sprachfunktionen verwenden.

Wenn Sie für ein Projekt „Nur Spark“ angeben, sind in Flex 3 verfügbare MX-Komponenten nicht für Anwendungen im Projekt verfügbar.

1 Wählen Sie „Datei“ > „Neu“ > „Flex-Projekt“.

2 Geben Sie den Projektpfad und die Servereinstellungen wie unter „Flex-Projekte“ auf Seite 90 beschrieben an.

3 Geben Sie auf der Seite „Erstellungspfad“ des neuen Assistenten „Neues Flex-Projekt“ „Nur Spark“ an.

4 Geben Sie weitere Erstellungspfadeigenschaften wie unter „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96 beschrieben an. Klicken Sie auf „Fertig stellen“.

Flex-Mobilprojekte

Mit Flash Builder können Sie Flex-Projekte erstellen, die für Mobilgeräte vorgesehen sind. In einem Flex-Mobilprojekt wird eine AIR-Anwendung erstellt, die auf Apple iOS-, BlackBerry- Tablet OS- und Google Android-Geräten ausgeführt werden kann.

Flex-Mobilprojekte erstellen

Mit diesem Vorgang können Sie ein Flex-Mobilprojekt für Apple iOS-, BlackBerry- Tablet OS- und Google Android-Geräte erstellen.

1 Wählen Sie „Datei“ > „Neu“ > „Flex-Mobilprojekt“.

2 Geben Sie einen Projektnamen und Projektpfad ein.

Der Standardspeicherort ist der aktuelle Arbeitsbereich.

3 Verwenden Sie das Standard-SDK von Flex 4.6, mit dem die Entwicklung von Mobilanwendungen möglich ist. Klicken Sie auf „Weiter“.

4 Geben Sie Mobileinstellungen an:

- Wählen Sie die Zielplattform für die Anwendung aus. Weitere Informationen finden Sie unter Zielplattformen auswählen.
- Geben Sie eine Anwendungsvorlage an.

  Weitere Informationen finden Sie unter Anwendungsvorlage auswählen.
- Legen Sie die Berechtigungen fest.

  Wählen Sie die Zielplattform und legen Sie die Berechtigungen für jede Plattform fest. Sie können die Berechtigungen später in der Anwendungsbeschreibungs-XML-Datei ändern.

  Weitere Informationen finden Sie unter Mobilanwendungsberechtigungen.
- Legen Sie die Plattformberechtigungen fest.

  In den Plattformeinstellungen könenn Sie eine Zielgerätefamilie auswählen. Je nach der ausgewählten Plattform können Sie das Zielgerät oder eine Zielgerätefamilie auswählen.

  Hinweis: *Es gibt keine plattformspezifischen Einstellungen für die Google Android- oder BlackBerry Tablet OS-Plattform.*

  Weitere Informationen finden Sie unter Plattformeinstellungen auswählen.
- Legen Sie die Anwendungseinstellungen fest.

  Weitere Informationen finden Sie unter Anwendungseinstellungen auswählen.

5 Klicken Sie auf „Fertig stellen“ oder auf „Weiter“, um die Servereinstellungen anzugeben.

6 (Optional) Geben Sie die Servereinstellungen an.

Weitere Informationen finden Sie unter „Auf Datendienste zugreifen“ auf Seite 221.

7 Geben Sie einen Ausgabeordner an.

Wenn Sie keinen Anwendungsserver angeben, befindet sich der Speicherort im Projektordner.

Wenn Sie einen Anwendungsserver angeben, befindet sich der Ausgabeserver außerhalb des Projektordners. Normalerweise wird der Ausgabeordner bei den Dienstdateien abgelegt.

8 Klicken Sie auf „Fertig stellen“ oder „Weiter“, um zusätzliche Konfigurationsoptionen anzugeben.

9 (Optional) Geben Sie Erstellungspfade und andere Konfigurationsoptionen ein.

Weitere Informationen finden Sie unter „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

10 Klicken Sie auf „Fertig stellen“.

Flex-Fachmann Brian Telintelo schrieb einen Artikel über die Erstellung einer Mobilanwendung für die Android-Plattform.

Weitere Informationen zur Entwicklung von Mobilanwendungen mit Flex und Flash Builder finden Sie unter Entwickeln von Mobilanwendungen mit Flex und Flash Builder.

Verwandte Themen

„Flex-Mobilprojekte“ auf Seite 35

ActionScript-Projekte

Mit ActionScript-Projekten können Sie Web- oder Desktopanwendungen erstellen, die entweder auf der Flash-API (nicht dem Flex-Framework) oder AIR-APIs basieren.

ActionScript-Projekte erstellen

Mit diesem Vorgang können Sie eine ActionScript-Anwendung für Web oder Desktop erstellen.

1 Wählen Sie „Datei“ > „Neu“ > „ActionScript-Projekt“.

2 Geben Sie einen Projektnamen und Projektpfad ein.

Der Standardspeicherort ist der aktuelle Arbeitsbereich.

3 Wählen Sie als Anwendungstyp entweder Web oder Desktop.

4 Geben Sie Erstellungspfade an.

Weitere Informationen finden Sie unter „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

5 Klicken Sie auf „Fertig stellen“.

Weitere Informationen zum Kodieren in ActionScript finden Sie im ActionScript-Entwicklerhandbuch.

ActionScript-Mobilprojekte erstellen

Verwenden Sie dieses Verfahren zum Erstellen von AIR-basierten ActionScript-Anwendungen, die auf Apple iOS-, BlackBerry Tablet OS- und Google Android-Geräten ausgeführt werden können.

1 Wählen Sie „Datei“ > „Neu“ > „ActionScript-Mobilprojekt“.

2 Geben Sie einen Projektnamen und Projektpfad ein.

Der Standardspeicherort ist der aktuelle Arbeitsbereich.

3 Verwenden Sie das Standard-SDK von Flex 4.6, mit dem die Entwicklung von Mobilanwendungen möglich ist. Klicken Sie auf „Weiter“.

4 Geben Sie Mobileinstellungen an:

- Wählen Sie die Zielplattformen für die Anwendung aus.

  Weitere Informationen finden Sie unter Zielplattformen auswählen.

- Legen Sie die Plattformberechtigungen fest.

  Wählen Sie die Zielplattform und legen Sie die Berechtigungen für jede Plattform fest. Sie können die Berechtigungen später in der Anwendungsbeschreibungs-XML-Datei ändern.

  Weitere Informationen finden Sie unter Mobilanwendungsberechtigungen.

- Legen Sie die Plattformberechtigungen fest.

  Weitere Informationen finden Sie unter Plattformeinstellungen auswählen.

- Legen Sie die Anwendungseinstellungen fest.

  Wählen Sie „Automatisch neu ausrichten“, wenn sich die Anwendung gleichzeitig mit dem Gerät drehen soll.

  Wählen Sie „Vollbild“, wenn die Anwendung im Vollbildmodus auf dem Gerät angezeigt werden soll.

5 Klicken Sie auf „Fertig stellen“ oder „Weiter“, um zusätzliche Konfigurationsoptionen anzugeben.

6 (Optional) Geben Sie Erstellungspfade an.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

7 Klicken Sie auf „Fertig stellen“.

Lesen Sie den Artikel zum Erstellen einer Mobilanwendung in ActionScript für die Android-Plattform vom Adobe Community-Experten Randy Troppmann.

Verwandte Themen

„ActionScript-Mobilprojekte“ auf Seite 36

„Startkonfigurationen verwalten“ auf Seite 123

Mit Flash Catalyst kompatible Projekte

Sie können ein Flex-Projekt erstellen, das mit Flash Catalyst kompatibel ist.

Beim Erstellen eines mit Flash Catalyst kompatiblen Projekts wird der Namespace `xmlns:fc="http://ns.adobe.com/flashcatalyst/2009"` hinzugefügt. Mit diesem Namespace können Sie Komponenten in Flash Builder erstellen, die in Flash Catalyst verwendet werden können. Flash Catalyst CS5.5 unterstützt einen Teil der in Flex 4.5 oder höher verfügbaren Komponenten.

Die Flash Catalyst-Kompatibilitätsprüfung wird für ein mit Flash Catalyst kompatibles Projekt automatisch aktiviert („Projekt“ > „Flash Catalyst“ > „Kompatibilität automatisch prüfen“). Wenn Sie inkompatible Elemente verwenden oder Kompatibilitätsprobleme im Projekt verursachen, zeigt Flash Builder die Kompatibilitätsfehler in der Problemansicht an. Weitere Informationen zu Richtlinien zum Erstellen eines mit Flash Catalyst kompatiblen Projekts finden Sie unter Richtlinien zum Erstellen eines mit Flash Catalyst kompatiblen Projekts.

Mithilfe eines mit Flash Catalyst kompatiblen Projekts können Designer und Entwickler an demselben Projekt zusammenarbeiten. Mithilfe von FXP- oder FXPL-Dateien können Sie ein mit Flash kompatibles Projekt sowohl in Flash Builder als auch in Flash Catalyst nutzen. Außerdem kann ein mit Flash Catalyst kompatibles Projekt direkt in Flash Builder bearbeitet werden. Verwenden Sie diesen Arbeitsablauf, wenn sowohl Flash Builder als auch Flash Catalyst auf demselben Computer installiert sind.

Weitere Informationen finden Sie unter „Verwenden von Flash Builder mit Flash Catalyst“ auf Seite 257.

Verwandte Themen

Workflows in Flash Catalyst und Flash Builder

Mit Flash Catalyst kompatible Flex-Projekte entwickeln

1 Wählen Sie „Datei“ > „Neu“ > „Mit Flash Catalyst kompatibles Projekt“ aus.

2 Geben Sie einen Projektnamen und Projektpfad ein.

Der Standardspeicherort ist der aktuelle Arbeitsbereich.

3 Verwenden Sie das standardmäßige Flex-SDK oder navigieren Sie zu einem anderen installierten SDK. Klicken Sie auf „Weiter“.

4 Klicken Sie auf „Fertig stellen“.

Mit Flash Catalyst kompatible Projekte auf Kompatibilität prüfen

Um sicherzustellen, dass das Projekt mit Flash Catalyst kompatibel bleibt, haben Sie folgende Möglichkeiten:

- Wählen Sie „Projekt“ > „Flash Catalyst“. Aktivieren Sie „Kompatibilität automatisch prüfen“.
- Wählen Sie „Projekt“ > „Eigenschaften“ > „Flash Catalyst“. Aktivieren Sie „Flash Catalyst-Kompatibilitätsprüfung für dieses Projekt automatisch ausführen“.
- Wählen Sie „Projekt“ > „Flash Catalyst“ > „Kompatibilitätsprüfung ausführen“.

Mit Flash Catalyst kompatible Flash-Projekte entwickeln

1 Wählen Sie im Paket-Explorer ein Projekt aus und klicken Sie auf „Projekt“ > „Flash Catalyst“ > „Projekt in Flash Catalyst bearbeiten“, um Flash Catalyst zu starten.

2 Beim Bearbeiten des Projekts in Flash Catalyst ist das Projekt in Flash Builder gesperrt, damit keine in Konflikt stehenden Änderungen auftreten.

3 Nachdem Sie das Projekt in Flash Catalyst bearbeitet haben, speichern Sie die Änderungen und beenden Sie Flash Catalyst.

Hinweis: *Wenn Sie bei der Bearbeitung eines mit Flash Catalyst kompatiblen Projekts Inkompatibilitäten einführen, zeigt Flash Builder die Kompatibilitätsfehler in der Problemansicht an. Klicken Sie mit der rechten Maustaste auf das Problem, um Informationen zum Kompatibilitätsfehler zu erhalten. Achten Sie darauf, dass vor dem Öffnen des Projekts in Flash Catalyst alle Kompatibilitätsfehler behoben sind.*

4 Um das Projekt zum Bearbeiten in Flash Builder zu öffnen, wählen Sie „Projekt“ > „Flash Catalyst“ > „Arbeit am Projekt fortsetzen“.

Wenn Sie diese Option wählen, werden Sie zum Speichern der mit Flash Catalyst vorgenommenen Änderungen aufgefordert. Dadurch wird das neu gespeicherte Projekt wieder in Flash Builder geöffnet. Beim Import der neuen Projektversion aus Flash Catalyst wird gleichzeitig die ursprüngliche, aus Flash Builder exportierte Projektversion gelöscht.

5 Sie können jederzeit beim Bearbeiten des Projekts in Flash Catalyst die Änderungen verwerfen und zu Flash Builder zurückkehren. Dazu wählen Sie „Projekt“ > „Flash Catalyst“ > „Projektbearbeitung in Flash Catalyst abbrechen“.

Wenn Sie diese Option wählen, werden mit Flash Catalyst am Projekt vorgenommenen Änderungen verworfen und die Originalversion des Projekts wird geöffnet. Bei der Originalversion des Projekts handelt es sich um die Version, die zuerst in Flash Catalyst geöffnet wurde. Bevor Sie die in Flash Catalyst geöffnete Projektversion löschen, können Sie die Änderungen unter einem anderen Projektnamen an einem anderen Speicherort speichern.

AIR-Projekte erstellen

1 Öffnen Sie Flash Builder.

2 Wählen Sie „Datei“ > „Neu“ > „Flex-Projekt“.

3 Geben Sie den Projektnamen ein.

4 In Flex werden AIR-Anwendungen als Anwendungstyp betrachtet. Es gibt zwei Anwendungstypen: eine Webanwendung, die in Adobe Flash Player ausgeführt wird, und eine Desktopanwendung, die in Adobe AIR ausgeführt wird. Wählen Sie als Anwendungstyp „Desktop“.

5 Wählen Sie gegebenenfalls die Servertechnologie aus, die Sie mit der AIR-Anwendung verwenden möchten. Wenn Sie keine Servertechnologie verwenden, wählen Sie „Keiner“ aus und klicken Sie dann auf „Weiter“.

6 Wählen Sie den Ordner aus, in dem die Anwendung gespeichert werden soll. Standardmäßig ist dies der Ordner „bin-debug“. Klicken Sie auf „Weiter“.

7 Ändern Sie gegebenenfalls den Quell- und den Bibliothekspfad und klicken Sie dann auf „Fertig stellen“, um das AIR-Projekt zu erstellen.

   Spielen Sie das Video zum Erstellen Ihrer ersten Desktop-Anwendung mit Flash Builder von Adobe-Guru James Ward ab.

Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen

Wenn Sie ein Flex-Projekt erstellen, können Sie seine Konfiguration anpassen. Alle zusätzlichen Konfigurationsschritte sind optional.

Hinweis: *Sie können auch nach Erstellung des Projekts dessen Konfiguration ändern. Rufen Sie im Quellmodus von Flash Builder die Optionen „Projekt“ > „Eigenschaften“ auf.*

Quellpfad Mit der Registerkarte „Quellpfad“ fügen Sie zusätzliche Quellordner zu einem Projekt hinzu. Sie können die Quellordner neu anordnen, den Speicherort von Ordnern bearbeiten und Ordner aus dem Quellpfad entfernen.

Hauptquellordner, Hauptanwendungsdatei, URL des Ausgabeordners Standardmäßig legt Flash Builder Quelldateien in einem Ordner namens „src“ in Ihrem Projekt ab. Der Standardname der MXML-Hauptanwendungsdatei ist der Name des Projekts. Sie können diese Standardeinstellungen beim Erstellen des Projekts ändern.

Wenn Sie ein Projekt erstellen, führt Flash Builder Anwendungsdateien mit einer Standard-URL aufgrund Ihrer Projekteinstellungen aus. Geben Sie eine Ausgabeordner-URL an, um die Standardeinstellungen zu überschreiben. Die Ausgabeordner-URL ist dann nützlich, wenn Sie die Anwendung auf dem Projektserver erstellen und aktivieren, aber sie auf einem Webserver debuggen. Geben Sie dann die Webserver-URL als Ausgabeordner-URL an. Beispiel: Wenn Sie http://myserver/test.swf als Ausgabeordner-URL angeben, wird eine Startkonfiguration mit dieser URL erstellt.

Siehe „Projektausgabeordner einrichten“ auf Seite 109 und „Anwendungen ausführen und debuggen“ auf Seite 123.

Bibliothekspfad Auf der Registerkarte „Bibliothekspfad“ können Sie die Framework-Verbindung und Erstellungspfadbibliotheken angeben.

- Komponentensatz

  Normalerweise machen Sie alle Komponenten verfügbar. In manchen Fällen geben Sie nur MX-Komponenten an. Siehe „Komponentensatz („MX + Spark“, „Nur Spark“ oder „Nur MX“)“ auf Seite 107.

- Framework-Verbindung

Anwendungsklassen für Flex 4.5 oder höhere Versionen des Flex-Frameworks nutzen standardmäßig dynamische Verknüpfungen. Die folgenden Optionen sind ebenfalls standardmäßig aktiviert:

- RSL-Digests überprüfen (Für Produktion empfohlen)
- Unbenutzte RSLs entfernen

  Hinweis: *Diese Option ist nicht für Flex-Frameworks, die älter als Flex 4.5 sind, verfügbar.*
- Beim Debuggen lokale SWF-Debug-RSLs (Runtime Shared Libraries) verwenden
- Bibliotheksreihenfolge automatisch basierend auf Abhängigkeiten ermitteln

Weitere Informationen finden Sie unter „Framework-Verbindung für Anwendungen“ auf Seite 108.

- Erstellungspfadbibliotheken

  Sie können Projektbibliotheken, SWC-Bibliotheksordner oder SWC-Dateien zum Erstellungspfad hinzufügen und aus ihm entfernen. Sie können auch die Erstellungspfadreihenfolge ändern.

  Verwenden Sie die Schaltfläche „Bearbeiten“, um den Speicherort hinzugefügter Bibliotheken oder Ordner zu ändern.

  Verwenden Sie die Schaltfläche „Flex SDK hinzufügen“, um das Standard-SDK für ein Projekt wiederherzustellen, wenn Sie das Flex SDK aus dem Erstellungspfad entfernt haben.

Native Erweiterungen Auf der Registerkarte „Native Erweiterungen“ können Sie ActionScript Native Extension-Dateien (ANE) hinzufügen, um native Plattformfunktionen in Ihrer Anwendung bereitzustellen.

Wichtig: *Sie können ActionScript-Erweiterungen nur für Mobil- und Desktopprojekte, die AIR 3.0 unterstützen, erstellen.*

Weitere Informationen finden Sie unter Einem Projekt native -Erweiterungen hinzufügen.

Projekte exportieren und importieren

Mit Flash Builder können Sie Projekte im FXP-Format, ZIP-Format und aus offenen Dateiordnern exportieren und importieren. Das FXP-Format ist ein Archivformat, das Projektordner, Dateien und Metadaten zum Projekt enthält. Flash Builder exportiert Flex-Projekte im FXP-Format und Flex-Bibliotheksprojekte im FXPL-Format. ActionScript-Projekte können nur als Archivdatei exportiert werden, und zwar normalerweise im ZIP-Format.

Durch das Importieren und Exportieren von Projekten können Sie Projekte sicher zwischen verschiedenen Computern und Benutzern übertragen.

Hinweis: *Sie können auch mithilfe des Eclipse-Exportassistenten Flex-Projekte und Flex-Bibliotheksprojekte im ZIP-Format (in Flex Builder 3) oder in anderen Archivformaten exportieren.*

Projekte exportieren

Flash Builder kann verschiedene Projekte zusätzlich zu Eclipse-Exportfunktionen exportieren.

Flex-Projekt oder Flex-Bibliotheksprojekt als FXP-Datei exportieren

Einige Flex-Projekte erfordern beim Import eine besondere Behandlung. Siehe „Projekte, die eine besondere Behandlung erfordern“ auf Seite 102

1 Wählen Sie in Flash Builder „Datei“ > „Flash Builder-Projekt exportieren“.

Sie können auch das Kontextmenü für ein Projekt im Paket-Explorer verwenden. Wählen Sie „Exportieren“ > „Flash Builder“ > „Flash Builder-Projekt“.

2 Wählen Sie im Assistenten „Flex-Projekt exportieren“ das Projekt aus, das exportiert werden soll.

Alle für den Export verfügbaren Projekte werden im Projekt-Popupmenü aufgeführt.

3 Navigieren Sie zu dem Speicherort in Ihrem Dateisystem, in den Sie die FXP-Datei exportieren möchten.

4 (Optional) Wählen Sie „Projektkompilierung validieren“.

Vergewissern Sie sich mit dieser Option, dass Ihr Projekt ohne Fehler kompiliert wurde. Wenn Fehler vorliegen, können Sie Ihr Projekt dennoch exportieren.

5 Klicken Sie auf „Fertig stellen“.

Bei Serverprojekten werden absolute Pfade zu Serverressourcen als Pfadvariablen gespeichert. Wenn Sie das Projekt später importieren, geben Sie Werte für die Pfadvariablen an.

ActionScript-Projekt im ZIP-Format (oder in einem anderen Archivformat) exportieren

1 Wählen Sie in Flash Builder „Datei“ > „Exportieren“ > „Andere“.

2 Wählen Sie im Exportassistenten „Allgemein“ > „Archivdatei“ und klicken Sie anschließend auf „Weiter“.

3 Wählen Sie das Projekt und die Dateien, die Sie exportieren möchten:

- Erweitern Sie im linken Bedienfeld das Projekt, um die einzuschließenden Projektordner anzugeben.
- Geben Sie im rechten Bedienfeld für jeden gewählten Ordner die einzuschließenden Dateien aus.

4 Wechseln Sie zu einem Speicherort, an dem das exportierte Projekt gespeichert werden soll, und geben Sie einen Dateinamen an.

5 Legen Sie die gewünschten Archivdateioptionen fest und klicken Sie auf „Fertig stellen“.

Projekte importieren

Flash Builder kann Flex-Projekte, Flex-Bibliotheksprojekte und ActionScript-Projekte, die zuvor aus Flash Builder exportiert wurden, importieren. Sie können mehrere Versionen desselben Flex-Projekts oder Flex-Bibliotheksprojekts importieren.

Flex-Projekte oder Flex-Bibliotheksprojekte importieren

Sie können ein Projekt mithilfe einer exportierten FXP-Datei oder durch Navigieren zu einem Ordner, der das Projekt enthält, importieren.

1 Wählen Sie im Flash Builder-Menü „Datei“ > „Importieren“.

Sie können Projekte auch im Paket-Explorer mithilfe des Kontextmenüs importieren.

2 (Projektordner) Wenn Sie ein Projekt aus einem vorhandenen Projektordner importieren, aktivieren Sie die Option „Projektordner“ und navigieren Sie zum Ordner, der das Projekt enthält.

3 (FXP-Datei) Wenn Sie ein Projekt mithilfe einer FXP-Datei importieren, aktivieren Sie die Option „Datei“ und navigieren Sie zum Speicherort der Datei.

Wenn eine FXP-Datei mehr als ein Projekt enthält, können Sie einzelne Projekte zum Importieren auswählen.

4 (Bibliotheksprojekt oder FXPL-Projekt) Wenn Sie ein Bibliotheksprojekt oder ein Catalyst FXPL-Projekt importieren, können Sie festlegen, ob der Inhalt in ein vorhandenes Projekt importiert werden soll.

5 (FXP-Datei) Wenn ein Projekt mit demselben Namen im Arbeitsbereich vorhanden ist, geben Sie die Importmethode folgendermaßen an:

- „Als neues Projekt importieren“: Flash Builder fügt dem Projektnamen einen numerischen Bezeichner hinzu. Vorherige Versionen des Projekts werden beibehalten.

  Geben Sie im Feld zum Extrahieren einen Speicherort an, an den die Datei extrahiert werden soll. Normalerweise ist dieser Speicherort ein Ordner in Ihrem Flash Builder-Arbeitsbereich, der einen Projektordner darstellt. Sie können einen neuen Projektordner angeben oder einen vorhandenen Projektordner überschreiben.

- „Vorhandenes Projekt überschreiben“: Wählen Sie das Projekt aus, das Sie überschreiben möchten. Die vorherige Version des Projekts wird permanent entfernt.

6 (Pfadvariablen) Wenn Sie ein Projekt importieren, das Pfadvariablen definiert, aktualisieren Sie die Pfadvariablen für das Projekt.

Für ColdFusion, PHP, ADEP Data Services oder andere Servertechnologien kompilierte Projekte greifen mithilfe von Pfadvariablen auf einen Webserver und Serverressourcen zu. Andere Projekte haben möglicherweise benutzerdefinierte Pfadvariablen.

Wählen Sie die einzelnen Pfadvariablen und geben Sie jeweils einen gültigen Wert ein.

7 (Schriftartverweise) Wenn Sie eine von Catalyst exportierte FXP-Datei importieren, kann das Projekt Schriftartverweise enthalten. Sie haben die Möglichkeit, Schriftartverweise aufzulösen.

Siehe „Schriftartverweise beim Importieren von Catalyst-Projekten auflösen“ auf Seite 102.

8 Klicken Sie auf „Fertig stellen“.

9 (PHP-Serverprojekte) Wenn Sie ein Projekt des Anwendungsservertyps PHP importieren, installieren Sie Zend oder aktualisieren Sie Ihre vorhandene Zend-Installation.

Das Zend-Dialogfeld führt Sie durch den Vorgang.

Hinweis: *Wenn Sie im Zend-Dialogfeld den Vorgang abbrechen, müssen Sie das Zend Framework manuell installieren oder aktualisieren. Sie können nur dann auf PHP-Dienste zugreifen, wenn das Zend Framework korrekt installiert und konfiguriert ist. Weitere Informationen zum Installieren und Konfigurieren Ihrer Zend Framework-Installation sowie zur Problembehandlung finden Sie im Abschnitt Installieren des Zend Framework.*

10 (Serverprojekte) Stellen Sie Dienste bereit.

a Platzieren Sie Dienste manuell unter dem Webstamm des Servers. Verwenden Sie dieselbe Ordnerstruktur, die im Originalprojekt verwendet wurde.

b Wählen Sie in der Ansicht „Daten/Dienste“ im Kontextmenü eines Diensts „Aktualisieren“.

Flex 3-Projekte importieren

Mithilfe des Kompatibilitätsmodus von Flex 3 können Sie ein Flex 3-Projekt in Flash Builder 4.6 importieren. In diesem Fall bleiben Namespaces und Komponenten von Flex 3 unverändert. Sie können jedoch die Vorteile des Compilers von Flex 4.6 nutzen.

Neue Dokumente, die im Kompatibilitätsmodus von Flex 3 erstellt wurden, verwenden MX-Komponenten und den folgenden Namespace:

```
mx="http://www.adobe.com/2006/mxml"
```

1 Wählen Sie in Flash Builder „Datei“ > „Flex-Projekt importieren“.

2 Navigieren Sie zur zuvor exportierten ZIP-Datei des Flex 3-Projekts oder suchen Sie nach dem Flex 3-Projektordner.

3 Klicken Sie auf „Fertig stellen“.

4 Stellen Sie sicher, dass im Dialogfeld „Flex SDK-Version wählen“ die Option Flex 4 SDK angegeben ist. Wählen Sie die Option zum Verwenden des Kompatibilitätsmodus von Flex 3.

5 Klicken Sie auf „OK“.

Weitere Informationen zur Verwendung von Flash Builder 4 für Ihre vorhandenen Flex 3-Projekte finden Sie im folgenden Adobe DevNet-Artikel: *Moving existing Flex projects from Flex Builder 3 to Flash Builder 4*.

ActionScript-Projekte importieren

Importieren Sie im Eclipseassistenten ein ActionScript-Projekt.

1 Wählen Sie in Flash Builder „Datei“ > „Importieren“ > „Andere“ > „Allgemein“ > „Archivdatei“.

Sie können auch mithilfe des Kontextmenüs im Paket-Explorer ein ActionScript-Projekt importieren.

2 Wählen Sie im Dialogfeld „Flex-Projekt importieren“ die ZIP-Datei aus, die Sie importieren möchten.

3 Klicken Sie auf „Fertig stellen“.

Projekte importieren, die mit dem Eclipse-Exportassistenten exportiert wurden

Wenn Sie ein Projekt haben, das mithilfe des Eclipse-Exportassistenten exportiert wurde, können Sie das Projekt mit dem Eclipse-Importassistenten importieren. Wählen Sie „Datei“ > „Importieren“ > „Allgemein“. Navigieren Sie anschließend zu dem Ihrem Projekt entsprechenden Format.

Weitere Informationen zum Importieren von Projekten finden Sie in der Eclipse-Dokumentation. Diese Dokumentation ist als Hilfe in den Eclipse-Import- und Exportassistenten verfügbar.

Wenn das Projekt Dienste enthält, die mit Flash Builder-Werkzeugen zum Zugreifen auf Datendienste erstellt wurden, müssen Sie die Dienste manuell hinzufügen. Kopieren Sie die Serverdateien im Diensteordner auf einen geeigneten Server. Bestimmen Sie in der Ansicht „Daten/Dienste“ den Ort des Dienstes mithilfe der Diensteigenschaften des Servers.

Wenn Sie ein PHP-Projekt exportiert haben, das das Zend Framework verwendet, muss das Zend Framework auf dem Zielserver installiert sein. Ändern Sie die Datei `amf-config.ini`, die das Zend Framework konfiguriert. Geben Sie für `zend_path` den absoluten Pfad zum Zend-Installationsordner an.

Weitere Informationen zum Installieren, Konfigurieren und zur Fehlerbehebung bei der Zend Framework-Installation finden Sie unter Installieren des Zend Framework.

Projekte in mehrere Arbeitsbereiche importieren

Beim Importieren eines Projekts wird das Projekt in einen Flash Builder-Arbeitsbereich importiert. Ein Projekt kann in mehrere Arbeitsbereiche importiert werden. In diesem Szenario befinden sich die Projektdateien an nur einem Speicherort auf einer Festplatte, aber die einzelnen Arbeitsbereiche verweisen auf sie. Änderungen, die Sie an einem Projekt vornehmen, werden auf alle Arbeitsbereiche angewendet.

Quelldateien in ein neues Projekt importieren

Wenn Ihr Dateisystem Quelldateien und Elemente enthält, die nicht Bestandteile eines Projekts sind, können Sie für diese Dateien ein Projekt erstellen.

1 Wählen Sie im Flash Builder-Menü „Datei“ > „Neu“ > *Projekt*.

Das Projekt kann ein Flex-Projekt, ein Flex-Bibliotheksprojekt oder ein ActionScript-Projekt sein.

2 Geben Sie im Projekterstellungsassistenten für die Einstellungen der Quell- und Ausgabeordner den korrekten Speicherort in Ihrem Dateisystem an.

Hinweis: *Sie können auch die vom Assistenten festgelegten standardmäßigen Speicherorte übernehmen und die Quelldateien an die vordefinierten Orte verschieben.*

Änderungen in einem Projekt vergleichen

Wenn Sie mehrere Versionen eines Projekts importieren, können Sie den Inhalt der Versionen vergleichen, kopieren oder zusammenführen. Sie können nur unterschiedliche Versionen desselben Projekts vergleichen.

1 Wählen Sie im Paket-Explorer eines der Projekte, die Sie vergleichen möchten.

2 Öffnen Sie das Kontextmenü des Paket-Explorers und wählen Sie „Projekt mit Version vergleichen“.

Die Vergleichsansicht wird aufgerufen. In dieser Ansicht können Sie das Projekt mit anderen Versionen des Projekts vergleichen.

3 Wählen Sie die Version für den Vergleich aus. Daraufhin wird der Eclipse-Vergleichseditor geöffnet.

4 Navigieren Sie im Vergleichseditor zu der Datei, die Sie vergleichen möchten, und wählen Sie im Kontextmenü „Inhaltsvergleich anzeigen“.

Im Vergleichseditor werden beide Versionen der Datei angezeigt und die Unterschiede werden hervorgehoben.

Mit den Optionen des Vergleichseditors können Sie Unterschiede in der Datei kopieren oder zusammenführen. Weitere Informationen zum Vergleichseditor finden Sie in der Eclipse-Dokumentation.

Unterstützung von Flash Catalyst-Projekten

Flash Builder bietet Unterstützung für Anwendungsentwickler, die Adobe® Flash® Catalyst™ einsetzen.

Flash Catalyst exportiert ein Projekt als FXP-Datei und Komponenten als FXPL-Datei. Die FXP- und FXPL-Dateien können dann zur Entwicklung in Flash Builder importiert werden. Bei FXP-Dateien ist das resultierende Projekt ein Flex-Webprojekt, das in Adobe Flash Player ausgeführt wird. Eine FXPL-Datei enthält eine Bibliotheksdatei. Sie können eine FXPL-Datei als Flex-Bibliotheksprojekt oder den Inhalt in ein vorhandenes Flex-Projekt importieren.

- Wählen Sie zum Importieren einer FXP- oder FXPL-Datei in Flash Builder im Menü „Projekt“ die Optionen „Flash Catalyst“ > „Flash Catalyst-Projekt importieren“.
- Wählen Sie zum Exportieren eines Flex-Projekts als Flash Catalyst-Projekt im Menü „Projekt“ die Optionen „Flash Catalyst“ > „Flash Catalyst-Projekt exportieren“. Beim Export des Flex-Projekts können Sie die Kompatibilität des Flex-Projekts mit Flash Catalyst überprüfen.

Sie können auf der Basis eines Flash Catalyst-Projekts ein Adobe AIR-Projekt erstellen. Importieren Sie die FXP-Datei für das Catalyst-Projekt in Flash Builder. Konvertieren Sie den Anwendungstyp des Projekts von „Web“ (wird im Flash Player ausgeführt) in „Desktop“ (wird in Adobe AIR ausgeführt). Siehe „Webanwendungsprojekte in Desktopanwendungen umwandeln“ auf Seite 132.

Weitere Informationen über die Arbeitsabläufe bei Flash Builder und Flash Catalyst finden Sie unter „Verwenden von Flash Builder mit Flash Catalyst“ auf Seite 257.

Schriftartverweise beim Importieren von Catalyst-Projekten auflösen

Beim Importieren eines mit Adobe Catalyst erstellten FXP-Projekts kann das importierte Projekt Verweise auf Schriftarten enthalten, die nicht auf Ihrem System vorhanden sind.

Der Importassistent bietet eine Option zum Beheben von Problemen bei Schriftartverweisen mithilfe von CSS. Wenn Sie diese Option wählen, importiert Flash Builder das Catalyst-Stylesheet `Main.css`. `Main.css` enthält Verweise auf die im Projekt verwendeten Schriftarten.

Falls durch die Schriftarten, die im Stylesheet referenziert werden, Kompilierungsfehler auftreten, ändern Sie die Verweise im Stylesheet auf in Ihrem System vorhandene Schriftarten.

Catalyst FXPL-Projekte enthalten keine Stylesheets. Beim Importieren einer FXPL-Datei versucht Flash Builder, Probleme mit Verweisen auf Schriftarten zu korrigieren. Falls Flash Builder keine entsprechende Schriftart auf dem Zielsystem findet, bleiben die ursprünglichen Schriftartverweise intakt. Bei FXPL-Projekten werden Schriftartverweise, die Flash Builder nicht auflösen kann, zur Laufzeit erkannt. Wenn keine Ersatzschriftart gefunden wird, tritt ein Laufzeitfehler für nicht aufgelöste Schriftartverweise auf.

Hinweis: *Bei FXPL-Dateien ändert Flash Builder das `fontFamily`-Attribut in MXML-Dateien, wenn das Programm versucht, Schriftartverweise aufzulösen.*

Projekte, die eine besondere Behandlung erfordern

Einige Flex-Projekte erfordern beim Import und Export eine besondere Behandlung. Beispiel:

- Das Projekt verweist auf eine frühere Version des Flex SDK
- Das Projekt verweist für den Zugriff auf serverseitige Daten auf Dienstdateien
- Die Zend Framework-Konfiguration für den Zugriff auf PHP-Dienste muss aktualisiert werden
- Das Projekt verwendet ADEP Data Services-Verknüpfungen zu einer Datenmodelldatei.

Einige Projektinhalte erfordern eine besondere Behandlung beim Exportieren bzw. Importieren eines Flex-Projekts.

- Unterschiedliche Versionen des Flex SDK

  Sie können Flex-Projekte importieren, die auf eine Flex SDK-Version, die in Ihrer Flash Builder-Installation nicht vorhanden ist, verweisen. Informationen zum Herunterladen und Installieren weiterer Flex SDK-Versionen finden Sie unter „Installierte Flex SDKs“ auf Seite 276. Wenn Flash Builder keine bestimmte Version des Flex SDK finden kann, navigieren Sie zum Speicherort des SDK.

- Dienstdateien

  Flex-Serverprojekte, die eine Verbindung zu Datendiensten wie beispielsweise ColdFusion oder BlazeDS herstellen, enthalten einen Ordner services, mit dessen ActionScript-Klassenreferenz Serverdateien bereitgestellt wurden. Beim Exportieren des Projekts wird der Ordner „services“ von Flash Builder exportiert, aber Sie müssen sicherstellen, dass ein Server und die entsprechenden serverseitigen Dateien beim Importieren vorhanden sind. Beim Importieren müssen Sie möglicherweise die serverseitigen Dateien und Aktualisierungen der Serveradressen in Klassen in den serverseitigen Dateien manuell bereitstellen. Für Projekte, die eine Verbindung zu Diensten mithilfe von ADEP Data Services oder BlazeDS herstellen, beachten Sie, dass die Dienstziele auf dem Zielserver verfügbar sind.

- Zend Framework

  Flex-Projekte, die mithilfe von PHP und dem Zend Framework mit Datendiensten verbunden werden, enthalten zwei Konfigurationsdateien. Überprüfen Sie diese Dateien beim Importieren, um sicherzustellen, dass sie für Ihr System korrekt konfiguriert sind:

  `amf-config.ini`

`gateway.php`

Siehe Installieren des Zend Framework

weitere Informationen zum Installieren, Konfigurieren und zur Fehlerbehebung in Ihrer Zend Framework-Installation.

- Datenmodelldateien (ADEP Data Services)

  Ein Flex-Projekt, das ADEP Data Services verwendet, wird mit einer Datenmodelldatei verknüpft. Nach dem Exportieren und anschließenden Importieren verweist Flash Builder auf die Datenmodelldatei selbst und nicht auf einen Link zu ihr. Wenn Sie statt der mit dem exportierten Projekt verpackten Datei eine verknüpfte Datei verwenden möchten, ändern Sie mithilfe der Projekteigenschaften die Datenmodelldatei. Wählen Sie „Projekt“ > „Eigenschaften“ > „Datenmodell“ und nehmen Sie die Änderungen vor.

Projekte erstellen

Adobe® Flash® Builder™ erstellt Ihre Projekte automatisch und exportiert diese dann in Anwendungen. Bei diesem Vorgang werden Anwendungs- und Bibliotheksdateien erzeugt, die Ausgabedateien werden im entsprechenden Ordner abgelegt und Sie werden über eventuelle Fehler, die während des Kompilierens aufgetreten sind, informiert.

Es gibt mehrere Möglichkeiten zur Änderung der Erstellungseinstellungen, um zu steuern, wie aus Projekten Anwendungen erzeugt werden. Sie können beispielsweise benutzerdefinierte Erstellungsvorgaben für einzelne oder alle Projekte im Arbeitsbereich festlegen. Außerdem können Sie den Erstellungsausgabepfad sowie die Erstellungsreihenfolge ändern usw. Sie können auch mit Werkzeugen von Drittanbietern wie z. B. Apache Ant benutzerdefinierte Erstellungsanweisungen erstellen.

Wenn die Anwendungen fertig sind, können Sie entweder alle oder ausgewählte Teile des Anwendungsquellcodes veröffentlichen. Benutzer können den Anwendungsquellcode in einem Webbrowser auf ähnliche Weise wie HTML-Quellcode anzeigen.

Anleitung zum Erstellen und Exportieren von Projekten

Ein typischer Arbeitsablauf besteht darin, aus Ihren Flex- und ActionScript-Projekten mit aktivierter Option „Automatisch erstellen“ eine Anwendung zu erzeugen. Während des Entwicklungsprozesses zeigt Flash Builder Ihnen in der Problemansicht Fehler und Warnungen an. Wenn Sie die Anwendung ausführen, werden eine Debugversion der SWF-Datei sowie erforderliche Elemente (Assets) und ein HTML-Wrapper in den Projektausgabeordner (bin) abgelegt. Diese Anwendung enthält Debuginformationen und ist nur für Entwickler geeignet. Weitere Informationen zum Exportieren von Projekten finden Sie unter „Projekte exportieren und importieren“ auf Seite 97.

Wenn die Anwendung fertig ist, erstellen Sie über den Assistenten zum Exportieren von Releasebuilds eine optimierte Version der Anwendung in Releasequalität. Hierdurch wird die SWF-Datei im Ordner „bin-release“ gespeichert. Da die Debugdaten entfernt wurden, ist die Datei kleiner. Diese Version ist ein Production-Build, das Endbenutzer anzeigen können. Bei Adobe AIR-Projekten werden AIR-Anwendungen in eine AIR-Datei exportiert. Mit der Option „Releasebuild exportieren“ erstellen Sie eine digital signierte AIR-Datei, die die Benutzer installieren müssen, bevor sie eine Anwendung ausführen können (ähnlich wie bei einer install.exe).

Bei Bibliotheksprojekten brauchen Sie nichts zu exportieren. Die von einem Flex-Bibliotheksprojekt erzeugte SWC-Datei ist sowohl für den Entwickler- als auch den Produktionseinsatz geeignet. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 234.

Grundlagen zum Erzeugen von Anwendungen aus Quelltext

MXML und ActionScript 3.0 sind *kompilierte* Sprachen. Kompilierte Sprachen unterscheiden sich von *interpretierten* Sprachen wie JavaScript, die von ihren Laufzeitumgebungen ausgeführt werden können. Das heißt, MXML und ActionScript 3.0 müssen zunächst in ein kompiliertes Format konvertiert werden, bevor sie von Flash Player ausgeführt werden können. Dieser Prozess, zusammen mit der Erzeugung der dazugehörigen Ausgabedateien, wird als *Erstellung* (englisch: Building) bezeichnet.

Flash Builder erstellt Ihre Projekte automatisch, wenn eine Datei im Projekt geändert und gespeichert wird. Sie können Anwendungen auch manuell erstellen. Ein Verständnis des Erstellungsprozesses und der Ausgabedateien, die erzeugt werden, hilft Ihnen bei der Diagnose und Behebung eventuell auftretender Projektkonfigurationsprobleme.

Flex-Projekte Quelldateien und eingebettete Elemente (z. B. Bilder) werden in eine einzige SWF-Ausgabedatei kompiliert. Die SWF-Datei kann direkt im eigenständigen Flash Player oder in einem Webbrowser durch eine *HTML-Wrapper*-Datei, die ebenfalls beim Erstellungsprozess erzeugt wird, ausgeführt werden. Diese Dateien werden im Ausgabeordner des Projekts erzeugt. Der Ausgabeordner wird standardmäßig mit „bin“ bezeichnet, aber Sie können diesen Ordner auch umbenennen.

ActionScript 3.0-Projekte Bei ActionScript 3.0-Projekten werden genau wie bei Flex-Projekten Quelldateien und eingebettete Elemente in eine SWF-Datei kompiliert.

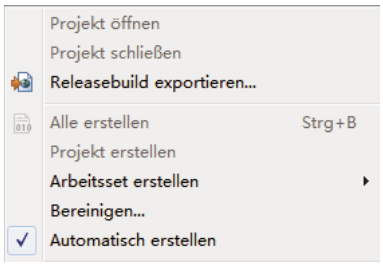
Flex-Bibliotheksprojekte Bei Bibliotheksprojekten sind Komponenten und dazugehörige Ressourcen die Quelldateien. Wenn Bibliotheksprojekte erstellt werden, wird im Ausgabeordner eine SWC-Datei erzeugt. In einer SWC-Datei ist eine SWF-Datei archiviert, die Komponenten, Ressourcen und eine Datei mit dem Namen „catalog.xml“ (das Manifest der in der SWF-Datei enthaltenen Elemente) enthält.

Automatischer Erstellungsprozess

In der eigenständigen Konfiguration von Flash Builder ist die Option „Automatisch erstellen“ standardmäßig aktiviert und die Anwendungen werden automatisch erstellt. Wählen Sie in der Plug-In-Konfiguration die Option „Automatisch erstellen“ aus. Wenn die Option „Automatisch erstellen“ deaktiviert wird, kann der Compiler keine Syntaxfehler erkennen. Die Problemansicht zeigt dann keine Warnungen und Fehlermeldungen an, während Sie Code eingeben. Nur wenn Sie das Projekt kompilieren, werden in der Problemansicht Warnungen und Fehlermeldungen angezeigt. Es empfiehlt sich deshalb Flash Builder so einzurichten, dass Anwendungen automatisch erstellt werden.

Erweiterte Optionen zum Erstellen von Projekten

Mit den erweiterten Erstellungsoptionen können Sie das Timing und den Umfang der erstellten Anwendungen steuern. Beispielsweise können Sie aus einem einzelnen Projekt oder aus allen Projekten im Arbeitsbereich einen Build erstellen, oder ein aus Projekten bestehendes Arbeitsset (eine Sammlung) anlegen, aus dem dann eine Anwendung erstellt wird. Alle Befehle zum Erstellen von Builds werden über das Menü „Projekt“ aufgerufen, wie im folgenden Beispiel gezeigt.



Der Flash Builder-Compiler ist inkrementell. Er verwendet im Erstellungsprozess nur die Ressourcen, die hinzugefügt und geändert wurden, und ignoriert alle anderen. Hierdurch werden Zeit und Systemressourcen gespart. Sie haben aber die Möglichkeit, aus allen Ressourcen im Projekt erneut einen Build zu erstellen. Hierzu würden Sie aus dem ganzen Projekt eine *neue Anwendung erstellen*. Dieser Schritt kann beispielsweise dann sinnvoll sein, wenn sich die Anwendung beim Testen unerwartet verhält und Sie alle potenziellen Fehlerquellen beseitigen möchten, indem Sie aus allen im Projekt enthaltenen Dateien eine neue Anwendung erstellen. Weitere Informationen hierzu finden Sie unter „Erweiterte Erstellungsoptionen“ auf Seite 111.

Wenn Sie im Arbeitsbereich Abhängigkeiten zwischen separaten Projekten erstellen, ermittelt der Compiler automatisch die Reihenfolge, in der die Projekte erstellt werden, damit diese Abhängigkeiten richtig aufgelöst werden. Sie können allerdings auch die vorgegebene Erstellungsreihenfolge außer Kraft setzen und die Reihenfolge, in der aus den Projekten im Arbeitsbereich eine Anwendung erstellt werden soll, manuell festlegen.

Außerdem besteht die Möglichkeit, den Erstellungspfad, die Anwendungsliste und Compilereinstellungen für jedes Projekt im Arbeitsbereich zu ändern.

Verwandte Themen

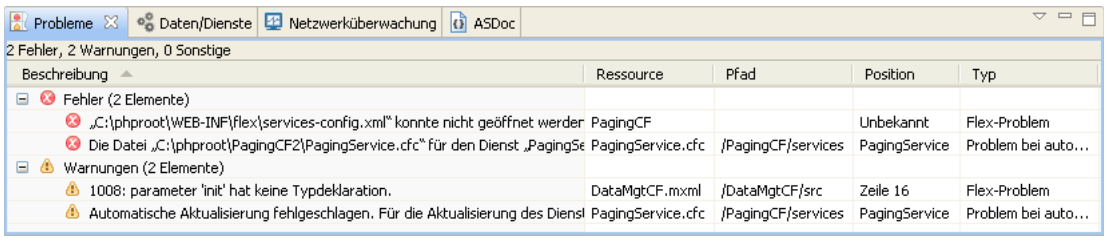
„Projekte manuell erstellen“ auf Seite 111

„Projektanwendungsdateien verwalten“ auf Seite 39

„Erweiterte Erstellungsoptionen“ auf Seite 111

In der Problemansicht angezeigte Erstellungsfehler

Fehler, die vom Compiler während des Erstellungsprozesses gefunden werden, werden in der Problemansicht angezeigt. Die Problemansicht ist in der Entwicklungs- und der Debugperspektive sowie im Codeeditor enthalten. Codezeilen, die Fehler enthalten, sind wie im folgenden Beispiel gezeigt mit einem x markiert:



Verwandte Themen

„Problemansicht“ auf Seite 11

Eclipse-Umgebungsfehler in der Logdatei

Manchmal sehen Sie Fehler, die von der Eclipse-Umgebung zurückgegeben werden. Diese Fehler treten am häufigsten dann auf, wenn Ressourcen wie z. B. SWC-Dateien nicht zur Laufzeit gefunden werden. In diesen Fällen werden die Fehlermeldungen in der Eclipse-Fehlerprotokolldatei angezeigt. Der Standardordner, in dem diese Logdatei unter Windows gespeichert ist, ist C:\Dokumente und Einstellungen*Benutzername*\workspace\.metadata\.log. Unter Mac OS befindet sich der Standardpfad ebenfalls im Workspace-Ordner, aber Dateien und Ordner, die mit einem Punkt beginnen, sind standardmäßig ausgeblendet.

Benutzerdefinierte Erstellungsskripten mit Apache Ant

Sie können den Standard-Erstellungsprozess ändern und erweitern, indem Sie Apache Ant (ein auf Java basierendes Open-Source-Erstellungswerkzeug) verwenden. Weitere Informationen zum Erstellen benutzerdefinierter Erstellungsprogramme finden Sie unter „Builds mit Apache Ant anpassen“ auf Seite 113.

Flex-Compileroptionen

Sie können die von Flash Builder verwendeten standardmäßigen Flex-Compilereinstellungen ändern. Öffnen Sie die Flex Compiler-Eigenschaftenseite, um die Standardeinstellungen anzuzeigen und zu ändern. Wählen Sie im Flash Builder-Menü „Projekt“ > „Eigenschaften“ > „Flex Compiler“.

Flex SDK-Version

Das Standard-SDK für Flash Builder ist Flex 4.6. Wenn Ihr Projekt jedoch eine bestimmte Version verwendet (z. B. Flex 3.5), kompiliert Flash Builder Anwendungen im Projekt mithilfe des angegebenen Flex SDK.

Sie können die Standardeinstellung ändern, sodass ein bestimmtes Flex SDK verwendet wird oder die Kompilierung mithilfe der Flex 3-Kompatibilität erfolgt. Das Angeben der Rückwärtskompatibilität beeinflusst in gewissem Maß das Verhalten der Anwendung, z. B. bezüglich der Layoutregeln, Auffüllung und Lücken, Skins und sonstige Stileinstellungen. Ferner wirkt sich dies auf die Regeln zum Analysieren der Eigenschaftendateien aus. Durch Definieren der Kompatibilitätsversion werden nicht alle existierenden Unterschiede zwischen den beiden Versionen erzwungen.

Adobe Flash Player-Optionen

Die standardmäßig vom Compiler verwendete Flash Player-Version ist die mindeste vom Flex SDK für die Kompilierung erforderliche Version.

Sie können eine bestimmte Version von Flash Player als Ziel für die Anwendung angeben. Funktionen, die eine höhere Version von Flash Player erfordern, werden nicht in die Anwendung kompiliert

Compileroptionen

Flash Builder enthält Kontrollkästchen für die folgenden Compileroptionen:

- Flash-Textwiedergabefunktion in MX-Komponenten verwenden

  Die Flash-Textwiedergabefunktion (Flash Text Engine, FTE) ist eine Bibliothek, die Textsteuerelemente mit zahlreichen Formatierungsoptionen bietet. Alle Spark-Komponenten im Paket spark.components unterstützen FTE. Siehe Schriftarten einbetten.

  Einige MX-Steuerelemente bieten Unterstützung für FTE. MX-Steuerelemente, die FTE unterstützen, verwenden dieselben eingebetteten Schriftarten wie Spark-Komponenten, die FTE verwenden. Siehe Using FTE in MX controls.

- Nicht eingebettete Dateien in den Ausgabeordner kopieren
- Barrierefreie SWF-Datei generieren

  Aktiviert beim Kompilieren der Anwendung oder SWC-Datei Funktionen für eine barrierefreie Bedienung. Informationen zum Verwenden von Barrierefreiheitsfunktionen bei Flex finden Sie im Abschnitt Accessible applications.

- Strict-Type-Prüfung aktivieren

Wenn „Strict-Type-Prüfung aktivieren“ aktiviert ist, druckt der Compiler undefinierte Eigenschafts- und Funktionsaufrufe aus. Außerdem führt der Compiler zur Kompilierzeit Typüberprüfungen von Zuordnungen und Optionen aus, die an Methodenaufrufe übergeben werden.

- Warnungen aktivieren

  Diese Option aktiviert angegebene Warnungen. Weitere Informationen finden Sie im Abschnitt Viewing warnings and errors.

Ferner können Sie mehrzeilige Compiler-Argumente angeben, die im Befehlszeilen-Compiler „mxmlc“ verfügbar sind. Sie können die Werte der meisten Optionen im Feld „Zusätzliche Compiler-Argumente“ festlegen, indem Sie dieselbe Syntax wie in der Befehlszeile verwenden. Weitere Informationen über die Syntax zum Festlegen von Optionen im Dialogfeld „Flex-Compiler“ finden Sie im Abschnitt About the command-line compilers.

Im Feld „Zusätzliche Compiler-Argumente“ können Sie einen Pfad zum SDK-Ordner ersetzen, indem Sie wie im folgenden Beispiel das `${flexlib}`-Token verwenden:

```
-include-libraries "${flexlib}/libs/automation.swc" "${flexlib}/libs/automation_agent.swc"
```

HTML-Wrapper

Der Flash Builder-Compiler generiert neben SWF-Dateien für Webanwendungen auch einen HTML-Wrapper, den Sie beim Bereitstellen der Anwendung verwenden können. Die folgenden Optionen stehen zur Auswahl:

- HTML-Wrapper-Datei generieren
- Ziel-Player-Version überprüfen

  Wenn diese Option aktiviert ist, überprüft die kompilierte Anwendung, ob die korrekte Version von Flash Player vorhanden ist.

  Ist die Option „Express-Installation verwenden“ aktiviert, führt die Anwendung im vorhandenen Flash Player eine SWF-Datei aus, um die aktuelle Version des Players beim Benutzer zu installieren.

- Integration mit Browsernavigation aktivieren

  Diese Option aktiviert tiefe Verlinkung Mithilfe von tiefer Verlinkung können Benutzer bei der Interaktion mit der Anwendung die Vor- und Zurück-Schaltflächen des Browsers verwenden.

Befehlszeilenzugriff auf die Flex-Framework-Compiler

Sie können die Flex-Framework-Compiler (mxmlc und compc) direkt über die Befehlszeile aufrufen. Weitere Informationen finden Sie unter Einführung in die Befehlszeilen-Compiler in Verwenden von Adobe Flex 4.6.

Projekt-Builds anpassen

Flash Builder ermöglicht es Ihnen, Anwendungen automatisch mit den Standardprojekteinstellungen zu erstellen. Das Verwenden der standardmäßigen Projekteinstellungen ist die empfohlene Vorgehensweise zum Erstellen von Anwendungen. Sie können aber die Projekterstellung auch entsprechend Ihren Bedürfnissen anpassen. Beispielsweise können Sie den Standardausgabeordner oder die Compileroptionen ändern.

Komponentensatz („MX + Spark“, „Nur Spark“ oder „Nur MX“)

Bei Flex-Projekten ist standardmäßig der gesamte Komponentensatz für Anwendungen im Projekt verfügbar. Zur Komponentengruppe gehören Spark-Komponenten, die mit Flex 4 eingeführt wurden, sowie MX-Komponenten, die in Flex 3 verfügbar waren.

In manchen Szenarien empfiehlt sich jedoch die ausschließliche Verwendung von MX-Komponenten, die in Flex 3 verfügbar waren, oder von Spark-Komponenten, die mit Flex 4 eingeführt wurden. Angenommen, Sie arbeiten an einem vorhandenen Flex 3-Projekt und möchten die neuen Spark-Komponenten nicht einführen. Sie möchten aber die Vorteile der Funktionen von Flex 4 und Flash Builder 4 nutzen, wie z. B. die neue Statussyntax, Verbesserungen des Compilers und Funktionen für andere Sprachen. In diesem Szenario wählen Sie den Komponentensatz „Nur MX“. Wenn Sie den Komponentensatz „Nur MX“ wählen, werden alle mit Spark in Verbindung stehenden Bibliotheken vom Erstellungspfad entfernt. Wenn Sie ein Flex 4-Projekt in „Nur MX“ konvertieren, ändert Flash Builder den Code in Ihrem Projekt nicht. Sie müssen Ihren Code manuell aktualisieren, um Verweise auf Spark-Komponenten und Bibliotheken zu entfernen.

Genauso gilt auch: Wenn Sie den Komponentensatz „Nur Spark“ wählen, werden nur die Spark-Komponenten verwendet und alle mit MX verbundenen Bibliotheken werden aus dem Erstellungspfad entfernt.

Framework-Verbindung für Anwendungen

Anwendungsklassen für Flex 4 oder höhere Versionen des Flex-Frameworks nutzen standardmäßig dynamische Verbindungen. Anstatt alle Klassen in die SWF-Anwendungsdatei zu kompilieren (statische Verbindung), werden einige Klassen aus der Framework-RSL (Runtime Shared Library) geladen. Anwendungen, die mit dynamischer Verbindung erstellt wurden, haben kleinere SWF-Dateien und werden folglich schneller heruntergeladen. Da aber alle Framework-Klassen geladen werden (nicht nur die Klassen, die Sie benötigen), belegen diese Anwendungen mehr Speicherplatz. Weitere Informationen hierzu finden Sie unter Runtime Shared Libraries.

Sie können die Eigenschaften eines Projekts ändern, um dieses Verhalten für alle in einem Projekt enthaltenen Anwendungen anzupassen. Wählen Sie hierzu das gewünschte Projekt aus und wählen Sie dann im Flash Builder-Menü „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“ > „Bibliothekspfad“ aus.

Flash Builder verwendet für die Framework-Verbindung standardmäßig das Standardverhalten des Flex SDK. Für Flex 4 und höher ist das Standardverhalten die dynamische Verbindung zu RSLs. Für Flex 3 ist das Standardverhalten die statische Verbindung. Sie können das Standardverhalten über die Dropdownliste „Framework-Verbindung“ ändern.

Für Flex 4.5 SDK und höhere Versionen des Flex-Frameworks sind die folgenden Optionen standardmäßig aktiviert:

- RSL-Digests überprüfen (für Produktion empfohlen)

  Überprüft, ob das Digest der RSL dem Digest entspricht, das zum Zeitpunkt der Kompilierung, als die Anwendung mit der Cross-Domain-RSL verbunden wurde, in der Anwendung gespeichert wurde. Weitere Informationen hierzu finden Sie unter About RSL digests.

- Unbenutzte RSLs entfernen

  Entfernt RSLs, die bei der Kompilierung nicht verwendet werden. Nur die verwendeten RSLs werden in den Ausgabeordner kopiert. Diese Option steht nur für Flex 4.5 SDK und höhere Versionen des Flex-Frameworks, nicht jedoch für ältere Versionen zur Verfügung.

  Sie können jedoch bei der Kompilierung das Laden einer unbenutzten RSL durch Doppelklicken auf den Projektbibliothekspfad erzwingen. Wählen Sie anschließend im Dialogfeld „Bibliothekspfadelementoptionen“ die Option „Laden von RSL erzwingen“.

  Sie können die Anwendungsdomäne angeben, in der die domänenübergreifenden RSLs geladen werden müssen. Sie können „Default“, „Current“, „Parent“ oder „Top-Level“ als Anwendungsdomäne auswählen.

  Wenn ein Modul beispielsweise eine bestimmte RSL verwendet, können Sie „Current“ als Anwendungsdomäne auswählen. Die angegebene RSL ist dann nur für dieses Modul zugänglich, jedoch nicht für die Anwendung, von der das Modul geladen wird. Durch das Entladen des Moduls wird automatisch auch die zugeordnete RSL entladen.

Hinweis: *Domänenübergreifende RSLs und Anwendungsdomänen werden nur in Flex 4.5 SDK und höheren Versionen des Flex-Frameworks, nicht jedoch in älteren Versionen unterstützt. Wenn Sie ein mit einem Flex-Framework erstelltes Projekt importieren, das Standard-RSLs verwendet, konvertiert Flash Builder die Standard-RSLs automatisch in domänenübergreifende RSLs.*

Weitere Informationen finden Sie unter Mit Standard- oder domänenübergreifende RSLs in Flash Builder kompilieren.

- Beim Debuggen lokale SWF-Debug-RSLs (Runtime Shared Libraries) verwenden

  Beim Debuggen der Anwendung werden lokale RSLs verwendet. Durch die Verwendung lokaler RSLs können Sie in den Debug-RSLs in bestimmte Codezeilen springen. Beim Exportieren einer Releasebuild wird diese Option ignoriert.

- Bibliotheksreihenfolge automatisch basierend auf Abhängigkeiten ermitteln

  Wenn diese Option aktiviert ist, bestimmt Flash Builder die Bibliotheksreihenfolge anhand der Abhängigkeiten in den Bibliotheken. Um eine andere Bibliotheksreihenfolge festzulegen, deaktivieren Sie diese Option und geben Sie die gewünschte Bibliotheksreihenfolge über die Pfeil-nach-oben- bzw. Pfeil-nach-unten-Schaltfläche an.

Automatische Builds aktivieren und deaktivieren

In der eigenständigen Konfiguration von Flash Builder werden Projekte automatisch erstellt. Wählen Sie in der Plug-In-Konfiguration diese Option selbst aus. Flash Builder ist so konzipiert, dass aus Projekten automatisch Anwendungen erstellt werden. Wenn Sie diese Option deaktivieren, kann der Compiler keine Syntaxfehler erkennen und auch keine Warnungen und Fehlermeldungen anzeigen, während Sie Code eingeben. Weitere Informationen zur manuellen Erstellung von Projekten finden Sie unter „Projekte manuell erstellen“ auf Seite 111.

Führen Sie einen der folgenden Schritte aus:

- Aktivieren Sie „Projekt“ > „Automatisch erstellen“.
- Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“. Aktivieren bzw. deaktivieren Sie die Option „Automatisch erstellen“.

Die Option „Automatisch erstellen“ wirkt sich auf alle Projekte im Arbeitsbereich aus.

Projektausgabeordner einrichten

Wenn Sie in Flash Builder ein Projekt erstellen, wird die erstellte Anwendung automatisch im Ausgabeordner generiert.

Sie können den Namen dieses Ordners ändern, wenn Sie das Projekt erstellen oder nachdem das Projekt erstellt ist. Sie können entweder einen Ordner erstellen oder einen vorhandenen Ordner im Arbeitsbereich auswählen.

1. Wählen Sie im Flex Paket-Explorer ein Projekt aus.
2. Klicken Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) und wählen Sie im Kontextmenü „Eigenschaften“.

   Das Dialogfeld mit den Projekteigenschaften wird angezeigt.
3. Wählen Sie die Eigenschaftenseite „Flex-Erstellungspfad“ aus.
4. Ändern Sie den vorhandenen Ausgabeordner, indem Sie einen neuen Namen eingeben, oder indem Sie zu einem vorhandenen Ordner im Projekt navigieren und ihn auswählen.

   Hinweis: *Der Ausgabeordner einer ADEP Data Services ES-Anwendung kann nicht auf diese Weise geändert werden, weil ihr Pfad über den Server gesteuert wird und nur über die Datei „Flex-config.xml“ aufrufbar ist.*
5. Klicken Sie auf „OK“.

Der neue Ausgabeordner ersetzt den vorhandenen Ausgabeordner.

Wichtig: *Wenn Sie den Namen des Ausgabeordners ändern, wird der ursprüngliche Ausgabeordner mitsamt seinem Inhalt gelöscht. Erstellen Sie aus dem Projekt erneut einen Build, um die SWF-Datei und die HTML-Wrapper-Dateien neu zu generieren.*

Projekterstellungspfad ändern

Jedes Projekt hat einen eigenen Erstellungspfad, der eine Kombination aus dem Quellpfad und dem Bibliothekspfad ist. (Erstellungspfade für Bibliotheksprojekte sind etwas komplexer. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 36.) Der Quellpfad ist der Pfad der MXML-Projektdatei und der ActionScript-Quelldateien. Der Bibliothekspfad ist der Pfad der grundlegenden Flex-Frameworkklassen sowie aller benutzerdefinierten Flex-Komponenten, die Sie in Form von SWC-Dateien erstellt haben.

Quellpfad ändern

1. Wählen Sie im Flex Paket-Explorer ein Projekt aus.
2. Klicken Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) und wählen Sie im Kontextmenü „Eigenschaften“. Das Dialogfeld mit den Projekteigenschaften wird angezeigt.
3. Wählen Sie die Eigenschaftenseite „Flex-Erstellungspfad“ aus. (Wenn Sie mit einem ActionScript-Projekt arbeiten, wählen Sie die Eigenschaftenseite „ActionScript-Erstellungspfad“ aus.)
4. Fügen Sie zum Quellpfad einen Ordner hinzu, indem Sie auf die Schaltfläche „Ordner hinzufügen“ klicken.
5. Geben Sie einen Namen für den Ordner ein oder klicken Sie auf die Schaltfläche „Durchsuchen“, um den Ordner mit den benutzerdefinierten Klassen auszuwählen.

   Sie können auch Pfadvariablen verwenden, statt den vollständigen Pfad zum Dateisystem einzugeben. Sie können entweder den Namen einer vorhandenen Pfadvariablen eingeben oder eine Pfadvariable erstellen. Weitere Informationen hierzu finden Sie unter „Pfadvariablen erstellen“ auf Seite 111.
6. Ändern Sie den Quellpfad wie gewünscht ab und klicken Sie auf „OK“.

Bibliothekspfad ändern

1. Führen Sie Schritt 1 bis 3 der zuvor beschriebenen Vorgehensweise aus, um die Eigenschaftenseite „Flex-Erstellungspfad“ aufzurufen.
2. Klicken Sie auf die Registerkarte „Bibliothekspfad“.

   Der Bibliothekspfad enthält Verweise auf die Flex-Frameworkklassen, die in SWC-Dateien enthalten sind. Eine SWC-Datei ist eine Archivdatei für Flex-Komponenten und andere Elemente. Weitere Informationen finden Sie unter „SWC-Dateien in Projekten verwenden“ auf Seite 237.

   Sie können den Pfad zum Framework bearbeiten oder (wenn Sie benutzerdefinierte Flex-Komponenten erstellt haben) neue Ordner oder SWC-Dateien zum Bibliothekspfad hinzufügen. Sie können auch Elemente aus dem Pfad entfernen.
3. Ändern Sie den Bibliothekspfad wie gewünscht ab und klicken Sie auf „OK“.

Pfadvariablen erstellen

Anstatt Ressourcen durch Eingabe des vollständigen Pfads zum lokalen Ordner oder Netzlaufwerk, auf dem Sie die Dateien speichern, zu verlinken, können Sie Pfadvariablen definieren. Beispiel: Sie definieren eine Pfadvariable mit dem Namen „Klassen“ und setzen dann den Pfad auf einen Ordner im Dateisystem. Anschließend wählen Sie „Klassen“ als Ort des neuen verlinkten Ordners. Wenn sich der Ordnerpfad ändert, können Sie die definierte Pfadvariable mit dem neuen Pfad aktualisieren. Alle Projekte, die mit „Klassen“ verlinkt sind, greifen so weiterhin auf die Ressourcen zu.

Pfadvariable festlegen oder erstellen

1. Wählen Sie im Flex Paket-Explorer ein Projekt aus.
2. Klicken Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) und wählen Sie im Kontextmenü „Eigenschaften“. Das Dialogfeld mit den Projekteigenschaften wird angezeigt.
3. Wählen Sie die Eigenschaftenseite „Flex-Erstellungspfad“ aus. (Wenn Sie mit einem ActionScript-Projekt arbeiten, wählen Sie die Eigenschaftenseite „ActionScript-Erstellungspfad“ aus.)
4. Sie können eine Pfadvariable für jedes Element im Pfad erstellen (einschließlich Ordner im Quellpfad und SWC-Ordner, Projekte und SWC-Dateien im Bibliothekspfad). Wählen Sie zum Beispiel auf der Registerkarte „Quellpfad“ die Schaltfläche „Ordner hinzufügen“. Das Dialogfeld „Ordner hinzufügen“ wird angezeigt.
5. Geben Sie eine Pfadvariable im folgenden Format ein: ${*pfadvariablenname*}.

   Hinweis: *Wenn die Variable nicht existiert, schlägt die Pfadeingabe fehl. Die Liste der vorhandenen Ressourcenvariablen kann über das Hauptmenü durch Auswahl von „Fenster“ > „Voreinstellungen“ und anschließende Auswahl von „Allgemein“ > „Arbeitsbereich“ > „Verlinkte Ressourcen“ aufgerufen werden. Auf dieser Eigenschaftenseite können Sie auch verlinkte Ressourcenvariablen verwalten.*
6. Klicken Sie auf „OK“, um die Pfadvariable zum Pfad hinzuzufügen.

Erweiterte Erstellungsoptionen

Flash Builder bietet erweiterte Optionen zum Anpassen der aus Projekten erstellten Anwendungen. Sie können beispielsweise Projekte manuell erstellen, die vorgegebene Erstellungsreihenfolge für Projekte im Arbeitsbereich ändern und mit dem Programm Apache Ant benutzerdefinierte Erstellungsprogramme erstellen.

Projekte manuell erstellen

Wenn Sie Projekte manuell erstellen, können Sie das Timing und den Umfang der erstellten Anwendung steuern. Sie können beispielsweise ein einziges Projekt oder alle Projekte im Arbeitsbereich erstellen, oder ein Arbeitsset mit Projekten oder ausgewählten Projektressourcen erstellen, um nur diese Projekte und Ressourcen zu erstellen. Ein Arbeitsset ist eine Sammlung von Arbeitsbereichressourcen (Projekte, Dateien und Ordner), die Sie auswählen und zusammenfassen können, um nur damit zu arbeiten. Weitere Informationen zu Arbeitssets finden Sie unter „Arbeitssets erstellen“ auf Seite 41.

Einzelnes Projekt erstellen

1. Wählen Sie im Flex Paket-Explorer das Projekt aus, das Sie erstellen möchten.
2. Wählen Sie im Hauptmenü „Projekt“ > „Projekt erstellen“ aus.

   Aus dem ausgewählten Projekt wird eine Anwendung erstellt und neue bzw. aktualisierte Release- und Debuganwendungsdateien werden zum Projektausgabeordner hinzugefügt.

Hinweis: *Wenn Projektdateien gespeichert werden müssen, werden Sie hierzu vom Programm aufgefordert, bevor der Erstellungsprozess beginnt. Wenn Sie diese Speicheraufforderung umgehen möchten, können Sie Arbeitsbereichvoreinstellungen festlegen, um Dateien vor dem Beginn eines Erstellungsprozesses automatisch zu speichern.*

Alle Projekte im Arbeitsbereich erstellen

❖ Wählen Sie im Hauptmenü „Projekt“ > „Alle erstellen“ aus.

Aus allen Projekten im Arbeitsbereich werden Anwendungen erstellt und jedem Projektausgabeordner werden Anwendungsdateien hinzugefügt. Sie werden anschließend zum Speichern von Dateien aufgefordert, sofern Sie vor dem Beginn des Erstellungsprozesses nicht die Option zum automatischen Speichern ausgewählt haben.

Arbeitsset erstellen

Führen Sie einen der folgenden Schritte aus:

- Wählen Sie im Hauptmenü „Projekt“ > „Arbeitsset erstellen“ > „Arbeitsset auswählen“. Klicken Sie auf „Neu“, um ein Arbeitsset zu erstellen. Weitere Informationen zum Erstellen eines Arbeitssets finden Sie unter „Arbeitssets erstellen“ auf Seite 41.
- Wählen Sie ein vorhandenes Arbeitsset, indem Sie im Hauptmenü „Projekt“ > „Arbeitsset erstellen“ > „Arbeitsset auswählen“ wählen.

Aus allen Projekten im Arbeitsset werden Anwendungen erstellt und die Anwendungsdateien werden zum Projektausgabeordner hinzugefügt.

Projektressourcen automatisch speichern

Wenn Sie aus Projekten manuell Anwendungen erstellen, werden Sie vor dem Beginn des Erstellungsprozesses aufgefordert, alle Ressourcen zu speichern. Um diese Aufforderung zu umgehen, können Sie Arbeitsbereichvoreinstellungen festlegen, damit Projektressourcen automatisch gespeichert werden.

1. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“.
2. Aktivieren Sie die Option „Vor Erstellung automatisch speichern“.
3. (Optional) Sie können ändern, wie oft Ressourcen gespeichert werden. Geben Sie hierzu in das Feld „Speicherintervall für Arbeitsbereich“ einen Wert (in Minuten) ein.

Neue Builds erstellen

Nachdem aus einem Projekt eine Anwendung erstellt wurde, sind bei den anschließenden Erstellungsprozessen nur die Ressourcen betroffen, die hinzugefügt oder geändert wurden. Um den Flash Builder-Compiler zu zwingen, alle Ressourcen in einem Projekt neu zu erstellen, können Sie einen neuen Build erstellen. Beispielsweise können Sie einen neuen Build erstellen, wenn Sie alle potenziellen Fehlerquellen eliminieren möchten, auf die Sie beim Testen der Anwendung gestoßen sind.

1. Wählen Sie im Hauptmenü „Projekt“ > „Bereinigen und neu erstellen“ aus.
2. Wählen Sie die Projekte aus, deren Builddateien Sie verwerfen und von Grund auf neu erstellen möchten.
3. Klicken Sie auf „OK“.

Erstellungsreihenfolge für Projekte ändern

Mit Flash Builder können Sie Beziehungen zwischen Projekten erstellen, wenn Sie im Arbeitsbereich mit mehreren Projekten arbeiten. Sie können beispielsweise ActionScript-Klassen aus einem Projekt in ein anderes importieren. Wenn Sie Beziehungen zwischen Projekten erstellen, wirkt sich dies auf die Reihenfolge aus, in der aus Projekten Anwendungen erstellt werden.

Standardmäßig erstellt der Compiler zusammengehörige Projekte in der für ihre richtige Erstellung erforderlichen Reihenfolge. Wenn ein Projekt beispielsweise auf Klassen in einem anderen Projekt verweist, wird aus dem Projekt, das die Klassen enthält, zuerst eine Anwendung erstellt. In den meisten Fällen reicht es aus, sich darauf zu verlassen, dass der Compiler die Projekte in der richtigen Reihenfolge erstellt, und die Anwendungen werden erfolgreich generiert.

Sie können die Erstellungsreihenfolge jedoch ändern. Beispielsweise können Sie die Erstellungsreihenfolge ändern, wenn Sie ein benutzerdefiniertes Erstellungsprogramm in Ant erstellt und es mit einem Projekt im Arbeitsbereich verbunden haben und dieses Projekt vor anderen Projekten erstellen müssen. Weitere Informationen zum Erstellen benutzerdefinierter Erstellungsprogramme finden Sie unter „Builds mit Apache Ant anpassen“ auf Seite 113.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“ > „Reihenfolge für Erstellung“ aus.

   Im Dialogfeld „Reihenfolge für Erstellung“ befinden sich die folgenden Optionen:

   Standarderstellungsreihenfolge verwenden Die Standarderstellungsreihenfolge ist durch die Abhängigkeiten zwischen Projekten vorgegeben und wird vom Compiler ausgeführt.

   Projekterstellungsreihenfolge Sie können die Erstellungsreihenfolge für alle Projekte im Arbeitsbereich manuell festlegen. Sie können auch ein Projekt aus der Erstellungsreihenfolgeliste entfernen. Das Projekt wird trotzdem erstellt, aber erst nachdem alle Projekte in der Erstellungsreihenfolgeliste erstellt wurden.

   Maximale Anzahl Iterationen bei Erstellung mit Zyklen Wenn Projekte zyklische Verweise enthalten (etwas, das Sie vermeiden sollten), können Sie die Anzahl der Erstellungsversuche festlegen, damit der Compiler alle Projekte richtig erstellen kann. Die maximal mögliche Anzahl Iterationen ist 10.

2 Ändern Sie die Erstellungsreihenfolge wie gewünscht ab und klicken Sie auf „OK“.

Builds mit Apache Ant anpassen

Wenn Sie ein benutzerdefiniertes Erstellungsprogramm erstellen, können Sie den Standarderstellungsprozess ändern und erweitern. Flash Builder enthält ein Standarderstellungsskript, das zum Kompilieren von Anwendungen verwendet wird. Bei Bedarf können Sie mit Apache Ant (ein auf Java basierendes Open-Source-Erstellungswerkzeug) benutzerdefinierte Erstellungsskripten erstellen.

Sie können benutzerdefinierte Erstellungsprogramme auf alle Flash Builder-Projektarten anwenden.

Erstellungsprogramm erstellen

1 Wählen Sie im Flex Paket-Explorer ein Projekt aus und klicken Sie dann mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste), um das Kontextmenü anzuzeigen. Wählen Sie dann „Eigenschaften“.

2 Wählen Sie die Eigenschaftenseite „Erstellungsprogramme“ aus. Wenn Sie andere Eclipse-Plug-Ins verwenden, sind u. U. mehrere Erstellungsprogramme aufgeführt. Flash Builder enthält ein Erstellungsprogramm namens Flex, das Sie nicht ändern können.

3 Wählen Sie „Neu“.

4 Wählen Sie im Dialogfeld „Konfigurationstyp auswählen“ den gewünschten Konfigurationstyp. Flash Builder unterstützt den Typ „Programm“. Wählen Sie den Typ aus und klicken Sie auf „OK“, um fortzufahren. Definieren Sie auf der neuen Eigenschaftenseite des Erstellungsprogramms die Erstellungsprogrammeigenschaften und referenzieren Sie das Ant-Skript (eine XML-Datei).

5 Klicken Sie auf „OK“, um das Erstellungsprogramm auf das Projekt anzuwenden.

Detaillierte Informationen zum Arbeiten mit Ant-Erstellungsskripten finden Sie in der Eclipse-Dokumentation unter help.eclipse.org/help31/index.jsp.

Mehrere SDKs in Flash Builder verwenden

In Flash Builder können Sie die Version des SDKs ändern, mit dem Projekte kompiliert werden. Sie können das SDK zu Beginn auswählen, wenn Sie ein Projekt erstellen, oder jederzeit, während Sie an einem Projekt arbeiten.

Das SDK besteht aus einem Framework und dem Compiler. Wenn Sie das Flex SDK 4 auswählen, verwenden Sie Version 4 der SWC-Dateien des Flex-Frameworks und Version 4 des Flex Compilers. Sie können beispielsweise nicht den Flex 4-Compiler mit SWC-Dateien des Flex-Frameworks 3 verwenden.

Die Verwendung eines anderen SDKs kann nützlich sein, wenn Sie ein Projekt erhalten, das mit Flex Builder 3 entwickelt wurde (das das Flex 3 SDK verwendet), aber mit Flash Builder 4 arbeiten (das standardmäßig das Flex 4 SDK verwendet). Durch die Auswahl eines älteren SDKs für den Erstellungsprozess können Sie Projekte pflegen, die nicht aktualisiert wurden und nicht mit der neuesten Version des SDKs kompatibel sind. Wenn Sie beispielsweise momentan an einem Projekt für das Flex 3 SDK arbeiten, aber Flash Builder 4-Funktionen nutzen möchten, können Sie Ihre Flash Builder-Version aktualisieren, aber dann das ältere SDK als Standard-SDK auswählen.

Wenn Sie ein Projekt entwickeln und dann das SDK ändern, erstellt Flash Builder das Projekt nicht inkrementell, sondern komplett neu. Hierbei weist Flash Builder auf alle Unterschiede hin, die Compilerfehler zurückgeben würden, genau so, als ob das Projekt mit dem ursprünglichen SDK entwickelt worden wäre.

Flash Builder generiert auch alle Hilfsdateien für Projekte neu. Hierzu gehören die Protokollverwaltungs- und tiefen Verlinkungsdateien, die vom HTML-Wrapper verwendet werden. Bei Flex 4 SDK-Projekten erstellt Flash Builder im Ordner „html-templates“ die mit dem Flex 4 SDK kompatiblen Protokollverwaltungsdateien „history.swf“, „history.html“ und „history.js“. Bei Flex 3 SDK-Projekten erstellt Flash Builder die mit dem Flex 3 SDK kompatiblen tiefen Verlinkungsdateien „deep-linking history.htm“, „history.js“ und „historyFrame.html“ im Ordner „html-templates/history“.

Zusätzlich ändern sich die in Flash Builder verfügbaren Optionen je nach dem ausgewählten SDK. Wenn Sie beispielsweise zu einem Projekt, das das Flex 3 SDK verwendet, ein Modul hinzufügen, lässt Flash Builder Sie nicht auswählen, ob Sie das Modul optimieren möchten oder nicht. Sie müssen dies manuell tun.

Weitere Informationen zu den Unterschieden zwischen dem Flex 4 SDK und dem Flex 3 SDK finden Sie unter Abwärtskompatibilität.

Wenn Sie ein Flex-Projekt erstellen, verwendet Flash Builder das Standard-SDK. Das Standard-SDK ist das neueste SDK, das mit Flash Builder ausgeliefert wird. Sie können aber auch jedes andere SDK auswählen, das in der Liste verfügbarer SDKs in Flash Builder angezeigt wird.

Wenn Sie ein Flex Bibliotheks- oder ActionScript-Projekt erstellen, können Sie im Dialogfeld „Neues FlexBibliotheksprojekt“ bzw. „Neues ActionScript-Projekt“ auswählen, welches SDK für dieses Projekt verwendet werden soll.

Neues Flex SDK zur Liste verfügbarer SDKs hinzufügen

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Installierte Flex SDKs“.

   Die momentan installierten SDKs werden in einer Liste angezeigt. Neben dem Namen des Standard-SDKs befindet sich ein Häkchen.

2 Klicken Sie auf „Hinzufügen“.

3 Geben Sie den Pfad des SDKs in das Feld „Flex SDK-Pfad“ ein.

4 Geben Sie einen Namen für das SDK im Feld „Flex SDK-Name“ ein. Verwenden Sie in diesem Feld keine Namen vorhandener SDKs.

5 Klicken Sie zum Speichern der Änderungen auf „OK“.

6 Klicken Sie erneut auf „OK“, um das neue SDK zur Liste verfügbarer SDKs hinzuzufügen. Diese Liste wird im Flash Builder-Arbeitsbereich für alle Flex-Projekte verwaltet. Wenn Sie ein neues Projekt erstellen, enthält die Liste verfügbarer SDKs dieses neue SDK.

SDK-Version für das aktuelle Projekt ändern

1 Wählen Sie „Projekt“ > „Eigenschaften“.

2 Wählen Sie „Flex Compiler“.

3 Aktivieren Sie die Option „Bestimmtes SDK verwenden“.

4 Wählen Sie das gewünschte SDK aus der Dropdownliste aus. Wenn das gewünschte SDK nicht in der Dropdownliste aufgeführt ist, klicken Sie auf den Link „Flex SDKs konfigurieren“.

5 Klicken Sie auf „OK“.

   Flash Builder wendet das neue SDK auf das aktuelle Projekt an. Fehler und Warnungen können zurückgegeben werden, wenn im Projekt Code verwendet wird, der nicht mit dem neuen SDK kompatibel ist.

Neues Standard-SDK auswählen

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Installierte Flex SDKs“.

   Neben dem Namen des Standard-SDKs befindet sich ein Häkchen.

2 Aktivieren Sie das Kontrollkästchen neben einem SDK. Dieses SDK wird das Standard-SDK. Es wird auch auf alle Projekte einschließlich des aktuellen Projekts angewendet, bei denen im Dialogfeld „Flex Compiler“ die Option „Standard SDK verwenden“ aktiviert ist. Wenn für das aktuelle Projekt ein bestimmtes SDK festgelegt wurde, wird dieses SDK auch dann weiterhin verwendet, wenn Sie das Standard-SDK des Arbeitsbereichs ändern.

3 Klicken Sie zum Speichern der Änderungen auf „OK“.

4 Klicken Sie erneut auf „OK“.

Alternativen zur Verwendung von Projektverweisen

Projektverweise können sich auf die Erstellungsreihenfolge auswirken. Flash Builder bietet daher Alternativen zur Verwendung von Projektverweisen.

Flex-Bibliotheksprojekte Die bevorzugte Methode zur Erstellung einer wiederverwendbaren Bibliothek. Flash Builder erstellt einen Projektverweis, um sicherzugehen, dass das SWC-Projekt vor dem Hauptprojekt, das es im Bibliothekspfad enthält, erstellt wird. Weil Flash Builder das Bibliotheksprojekt zum Bibliothekspfad hinzufügt, erscheinen im Hauptprojekt außerdem Codehinweise für die Klassen im SWC-Projekt.

Quellpfad Die empfohlene Methode, um in ein Projekt Code aufzunehmen, der sich nicht in derselben Ordnerstruktur befindet. Hierdurch erscheinen in den Projektdateien und den Klassen der zugehörigen Dateien

Codehinweise und der Compiler weiß, wo der Quellcode zu finden ist. Sie können beliebig viele Quellpfade zum Projekt hinzufügen. Die Quellpfade werden als verlinkte Ordner im Paket-Explorer angezeigt.

Flash Builder-Befehszeilenerstellung mithilfe von Apache Ant

Flash Builder stellt den Ant-Task <fb.exportReleaseBuild> zur Verfügung. Verwenden Sie diesen Task, um Befehlszeilenerstellungen zu implementieren, die die individuellen Erstellungseinstellungen eines Entwicklers mit dem nächtlichen Erstellungsprozess synchronisieren. Alternativ können Sie den Task <mxmlc> in benutzerdefinierten Skripten für nächtliche Erstellungsprozesse verwenden.

Hinweis: *Unterstützung für Befehlszeilenerstellung ist nur in Flash Builder Premium verfügbar.*

Task <fb.exportReleaseBuild>

Der Task <fb.exportReleaseBuild> gewährleistet, dass die Einstellungen des nächtlichen Erstellungsprozesses exakt den von Entwicklern während ihrer täglichen Arbeit verwendeten Einstellungen entsprechen.

Ändert ein Entwickler beispielsweise den Bibliothekspfad eines Flex-Projekts, so wird der neue Bibliothekspfad in das Flash Builder-Projekt geschrieben. Wenn der Rechner, auf dem nachts die Erstellung erfolgt, <fb.exportReleaseBuild> ausführt, lädt dieser Task dann das Flash Builder-Projekt sowie alle dazugehörigen Einstellungen.

Ein weiterer Vorteil, der sich bei Verwendung von <fb.exportReleaseBuild> ergibt, ist, dass automatisch Schritte ausgeführt werden, die normalerweise in einem Flash Builder-Erstellungsprozess enthalten sind, z. B.:

- Automatische Kompilierung verknüpfter Bibliotheksprojekte
- Kopieren von Elementen wie z. B. JPEG-Dateien in den Ausgabeordner
- Kopieren der HTML-Vorlage einschließlich Makroersetzung basierend auf den Kompilierungsergebnissen (z. B. Breite und Höhe)

Hinweis: *Um den Task <fb.exportReleaseBuild> verwenden zu können, müssen Sie Flash Builder auf dem Rechner installieren, auf dem nachts die Erstellung erfolgt.*

Task <mxmlc>

Wenn Sie ein benutzerdefiniertes Skript schreiben, das den Task <mxmlc> verwendet (z. B. ein Ant-Skript), brauchen Sie Flash Builder nicht auf dem Build-PC zu installieren. Auf dem Build-PC muss jedoch das Flex SDK vorhanden sein. Der Build-PC kann sich also auf einer Linux-Plattform befinden.

Der Nachteil dieses Ansatzes ist aber, dass Sie zwei Erstellungseinstellungssets synchronisieren müssen: einen in Flash Builder, der von den Entwicklern bei der täglichen Arbeit verwendet wird, und einen weiteren auf dem Rechner, auf dem die nächtliche Erstellung erfolgt.

Verwendung von <fb.exportReleaseBuild> verwenden

1. Installieren Sie Flash Builder auf einem Build-PC.
2. Schreiben Sie `build.xml` mit fb.exportReleaseBuild als Ziel. Beispiel:

```
<?xml version="1.0"?>
<project default="main">
    <target name="main">
        <fb.exportReleaseBuild project="MyProject" />
    </target>
</project>
```

`build.xml` gibt an, das Flex-Projekt über die Befehlszeile zu erstellen und hierzu die in den Projektdateien gespeicherten Einstellungen zu verwenden. Einzelheiten zu verfügbaren Parametern finden Sie unter „Parameter für den Task <fb.exportReleaseBuild>“ auf Seite 118.

3 Erstellen Sie ein Skript für einen nächtlichen Erstellungsprozess, das Eclipse anweist, nach einer Erstellungsdatei zu suchen und deren Ziel auszuführen.

Im folgenden Beispiel ist `build.xml` als Erstellungsdatei angegeben, die MyTarget ausführt.

Wenn Ihr nächtlicher Erstellungsprozess auf einer Macintosh-Plattform ausgeführt wird, können Sie das folgende Skript ausführen:

```
WORKSPACE="$HOME/Documents/Adobe Flash Builder"

# works with either FlashBuilder.app or Eclipse.app
"/Applications/Adobe Flash Builder/FlashBuilder.app/Contents/MacOS/FlashBuilder" \
    --launcher.suppressErrors \
    -noSplash \
    -application org.eclipse.ant.core.antRunner \
    -data "$WORKSPACE" \
    -file "$(pwd)/build.xml" MyTarget
```

Wenn Ihr nächtlicher Erstellungsprozess auf einer Windows-Plattform ausgeführt wird, können Sie die folgende Batchdatei ausführen:

```
set WORKSPACE=%HOMEPATH%\Adobe Flash Builder

REM works with either FlashBuilderC.exe or eclipsec.exe
"C:\Program Files\Adobe\Adobe Flash Builder 4.6\FlashBuilderC.exe" ^
    --launcher.suppressErrors ^
    -noSplash ^
    -application org.eclipse.ant.core.antRunner ^
    -data "%WORKSPACE%" ^
    -file "%cd%\build.xml" MyTarget
```

Eigenschaft <fb.running Ant>

Wenn Flash Builder ausgeführt wird, hat die Eigenschaft <fb.running Ant> den Wert „true“. Sie können diese Eigenschaft verwenden, wenn Sie Skripten in Flash Builder ausführen. Beispiel:

```
<target name="myFlashBuilderTasks" if="fb.running">
    <fb.exportReleaseBuild ... />
</target>
```

Ant-Tasks in Eclipse

Eclipse stellt mehrere Ant-Tasks zur Verfügung, die als Ziele in Erstellungsprozessskripten verwendet werden können. Beispiel:

- eclipse.incrementalBuild
- elicpse.refreshLocal
- eclipse.convertpath

Weitere Informationen zu diesen Skripts finden Sie in der Dokumentation zu Eclipse.

Parameter für den Task <fb.exportReleaseBuild>

Attribut	Beschreibung	Erforderlich?	Standardwert
application	Der Name der zu kompilierenden Anwendung. Hier braucht nur der Anwendungsname (ohne Pfad bzw. Erweiterung) angegeben zu werden. Beispiel: Anwendung1. Um Doppeldeutigkeiten bei Namen zu vermeiden, kann auch der vollständige Pfad relativ zum Projektstamm (z. B.: src/Anwendung1.mxml) angegeben werden. Um alle Anwendungen zu kompilieren, geben Sie „*“ an oder lassen Sie dieses Attribut weg. Wenn ein AIR-Projekt ausgeführt wird, kann nur eine einzige Anwendung angegeben werden. Der Wert „*“ ist in diesem Fall nicht erlaubt.	Nein	Die Standardanwendung des Projekts.
basefilename	Nur für Mobil- und AIR-Desktopprojekte: Der Name des zu exportierenden Pakets. Geben Sie den Paketnamen ohne die Dateinamenerweiterung an.	Ja	Der Name der ausgewählten Anwendung.
certificate	Für Mobil- und AIR-Projekte: Beim Exportieren einer AIR- oder nativen Installationsdatei der Pfad zum Zertifikat, das zum Signieren des Pakets verwendet wurde. Hinweis: Wenn dieser Wert weggelassen wird, wird der Wert des certificate-Attributs in der .actionScriptProperties-Datei des Projekts verwendet. Wenn in der .actionScriptProperties-Datei kein certificate-Attribut vorhanden ist, wird eine Fehlermeldung angezeigt.	Nein	–
destdir	Gibt den Ausgabeordner für den Export des Releasepakets an. Der Ordner kann ein relativer oder absoluter Pfad sein. Wenn Sie einen relativen Pfad angeben, ist der Pfad relativ zum Stamm des Projekts.	Nein	"bin-release" für Webprojekte und "" (gibt den Stamm des Projekts an) für alle anderen Projekttypen
failonerror	Gibt an, ob die Anwendungserstellung fehlschlagen soll, wenn Kompilierungsfehler auftreten.	Nein	true
locale	Gibt das Gebietsschema an, z. B. de-DE. Dieser Wert wird über den Compiler-Merker „-locale“ an den Compiler übergeben. Wenn dieses Attribut verwendet wird, wird statt des Gebietsschemas, das im Feld „Zusätzliche Compiler-Argumente“ angegeben wurde, dieser locale-Wert verwendet.	Nein	–
packagetype	Für Mobil- und AIR-Desktopprojekte: Gibt den Pakettyp an, der beim Exportieren des Releasepakets verwendet wird. Bei Mobilprojekten können Sie die Pakettypen „air“, „airi“ oder „platform“ angeben. Geben Sie „air“ an, wird eine einzige plattformunabhängige AIR-Datei generiert. Wenn Sie „airi“ angeben, wird für jede Plattform eine airi-Datei generiert. Wenn Sie „platform“ angeben, wird eine plattformspezifische Datei für jede Plattform generiert. Beispiel: Eine APK-Datei wird für die Android-Plattform und eine IPA-Datei für die iOS-Plattform generiert. Bei AIR-Desktopprojekten können Sie die Pakettypen „air“, „airi“ oder „native“ angeben. Bei jeder Option wird nur eine Datei generiert.	Ja	–

Attribut	Beschreibung	Erforderlich?	Standardwert
password	Nur für AIR-Projekte: Das Kennwort für das Zertifikat, das zum Signieren der AIR-Datei verwendet werden soll. Wenn dieses Argument weggelassen wird, wird eine Fehlermeldung angezeigt. **Vorsicht**: Die Angabe eines Literalwerts für ein Kennwort kann die Sicherheit gefährden.	Nein	–
Plattform	Nur für Mobilprojekte: Gibt die Zielplattform für den Export des Releasebuild an. Sie können für die Plattform „android", „ios" oder „qnx" angeben. Geben Sie mehrere Plattformen in einer durch Kommas getrennten Liste an. **Hinweis:** Bei Mobilprojekten mit dem Pakettyp „air" ist dieses Attribut nicht erforderlich.	Ja	–
Projekt	Das zu erstellende Projekt. Geben Sie den Namen eines Projekts im Flash Builder-Arbeitsbereich ohne den Pfad an. Beispiel: „MeinFlexProjekt".	Ja	–
publishsource	Gibt an, ob die Quelle der Anwendung veröffentlicht werden soll. Hierdurch können Benutzer Quelldateien über das Kontextmenü „Quelle anzeigen" anzeigen.	Nein	false
timestamp	Nur für AIR-Projekte: Gibt an, ob die erzeugte AIR-Datei mit einem Zeitstempel versehen werden soll. Hinweis: Wenn dieser Wert weggelassen wird, wird der Wert des timestamp-Attributs in der .actionScriptProperties-Datei des Projekts verwendet. Wenn in der .actionScriptProperties-Datei des Projekts kein timestamp-Attribut vorhanden ist, wird der Wert „false" verwendet.	Nein	false
verbose	Der Task <fb.exportReleaseBuild> gibt zusätzliche Informationen aus. Beispielsweise werden die Dateien aufgelistet, die in der AIR-Datei verpackt wurden, und es wird angegeben, wie lange jeder Prozessschritt gedauert hat.	Nein	false

Plattformspezifische Attribute für Mobilprojekte angeben

Durch die Angabe plattformspezifischer Attribute können Sie Releasepakete auf mehreren Plattformen gleichzeitig exportieren. Sie können plattformspezifische Attribute verwenden, indem Sie das plattformspezifische Präfix angeben. Wenn Sie beispielsweise das certificate-Attribut für die Android-Plattform verwenden möchten, geben Sie „android.certificate" an. Möchten Sie es für die iOS-Plattform verwenden, geben Sie „ios.certifcate" an.

Android-Plattform

Attribut	Beschreibung	Erforderlich?
android.airDownloadURL	Die URL zum Herunterladen der AIR-Laufzeitumgebung, wenn diese beim Start der Anwendung nicht auf dem Gerät des Benutzers installiert ist. **Hinweis:** Wenn Sie dieses Attribut nicht angeben, wird die standardmäßige Android Market-URL verwendet.	Nein
android.certificate	Der Pfad zum Zertifikat, das zum Signieren der APK-Datei verwendet wird. Sie legen dieses Attribut fest, wenn Sie für den „packagetype“ „platform“ angeben und eine der Zielplattformen Android ist.	Ja, wenn Sie in der .actionScriptProperties-Datei nicht den Zertifikatspfad für die Android-Plattform definiert haben.
android.password	Das Kennwort für das Zertifikat, das zum Signieren der APK-Datei verwendet wird. Wenn dieses Attribut weggelassen wird, wird eine Fehlermeldung angezeigt.	Ja

iOS-Plattform

Attribut	Beschreibung	Erforderlich?
ios.certificate	Der Pfad zum Zertifikat (Dateinamenerweiterung „.p12“), das zum Signieren der IPA-Datei verwendet wird. Sie legen dieses Attribut fest, wenn Sie für den „packagetype“ „platform“ angeben und eine der Zielplattformen Apple iOS ist.	Ja, wenn Sie in der .actionScriptProperties-Datei nicht den Zertifikatspfad definiert haben.
ios.packagetype	Der Pakettyp, der beim Exportieren einer IPA-Datei verwendet wird. Legen Sie für den Pakettyp bei Ad-hoc-Verpackung „adhoc“ und bei Verpackung für die Übermittlung an den Apple App Store „appstore“ fest.	Ja, wenn Sie in der .actionScriptProperties-Datei nicht den Pakettyp definiert haben.
ios.password	Das Kennwort für das Zertifikat, das zum Signieren der IPA-Datei verwendet wird. Wenn dieses Argument weggelassen wird, wird eine Fehlermeldung angezeigt.	Ja
ios.provisioning	Der Pfad zur Bereitstellungsdatei (Dateinamenerweiterung „.mobileprovision“), die zum Signieren der IPA-Datei verwendet wird.	Ja, wenn Sie in der .actionScriptProperties-Datei nicht den Pfad zur Bereitstellungsdatei definiert haben.

Assistent zum Exportieren von Releasebuilds

Wenn Sie den Assistenten zum Exportieren von Releasebuilds („Projekt“ > „Releasebuild exportieren“) ausführen, werden die von Ihnen im Assistenten vorgenommenen Einstellungen in der Datei „.actionScriptProperties“ gespeichert. Die Einstellungen des Assistenten werden von einem Befehlszeilenerstellungsprozess, der den Task <fb.exportReleaseBuild> verwendet, abgerufen. Der Assistent zum Exportieren von Releasebuilds speichert die folgenden Einstellungen:

- Quelle anzeigen

  Die von Ihnen für „Quelle anzeigen“ angegebenen Quelldateien werden gespeichert. Wenn Sie für den Parameter <publishsource> „fb.exportReleaseBuild“ angeben, schließt der Assistent diese Dateien als anzeigbare Quelldateien mit ein.

  Wichtig: *Bei Serverprojekten können Sie beim Exportieren von Quelldateien den Diensteordner festlegen. Das Exportieren von Dateien, die Dienste implementieren, führt zu Auswirkungen auf die Sicherheit. Diese Dateien ermöglichen unter Umständen den Zugriff auf Ihre Datenbank sowie auf Benutzernamen und Kennwörter. Siehe Exportieren von Quelldateien mit der Release-Version einer Anwendung.*

- Bei AIR-Projekten werden alle zusätzlichen Ausgabedateien, die Sie im Assistenten angeben, mit in die AIR- oder AIRI-Datei aufgenommen.

Befehlszeilenerstellungsprozesse auf Linux- und anderen Plattformen ausführen

Der Task <fb.exportReleaseBuild> wird nur unter Windows und Mac OS unterstützt.

Wenn Sie ein Erstellungsskript für eine andere Plattform schreiben, verwenden Sie die Option `-dump-config` für den mxmlc- oder compc-Compiler, um Compiler-Konfigurationseinstellungen in eine Datei zu schreiben. Anschließend können Sie die Konfigurationsoptionen mit der Option „-load-config“ auslesen.

Ändern Sie die Konfigurationseinstellungen in der Datei wie gewünscht ab. Ändern Sie beispielsweise <debug>true</debug> zu <debug>false</debug>, wenn im nächtlichen Erstellungsprozess ein Releasebuild erstellt werden soll.

Befehlszeilenerstellungsprozess mit Compiler-Einstellungen aus Flash Builder ausführen

1. Wählen Sie in Flash Builder die Option „Projekt“ > „Eigenschaften“ > „Flex Compiler“ aus.
2. Legen Sie unter „Zusätzliche Compiler-Argumente“ das folgende Argument fest:

   -dump-config *pfadname*, wobei *pfadname* den absoluten Pfad zu einer Datei im System angibt.
3. Übernehmen Sie die Änderungen im Projektfenster.

   Die Compiler-Einstellungen werden in die angegebene Datei geschrieben. Überprüfen Sie, ob die Datei geschrieben wurde. Wenn ja, entfernen Sie das Argument „-dump-config“.
4. Ändern Sie die Konfigurationseinstellungen wie gewünscht ab.
5. Führen Sie in Ihrem Erstellungsskript den Compiler aus, damit die gespeicherten Compiler-Einstellungen in das Skript aufgenommen werden:

   mxmlc -load-config *pfadname*

Einschränkungen bei Befehlszeilenerstellungsprozessen

Beim Ausführen von Erstellungsprozess über die Befehlszeile mit dem Task <fb.exportReleaseBuild> sind einige Einschränkungen zu beachten.

Befehlszeilenerstellungsprozesse auf 64-Bit-Plattformen ausführen

Flash Builder läuft auf Plattformen, die 32-Bit Java implementieren. Um einen Befehlszeilenerstellungsprozess auf Plattformen auszuführen, die 64-Bit Java unterstützen (z. B. Mac OS X Snow Leopard), müssen Sie den Befehlszeilenoptionen, die an Java übergeben werden, den Parameter `-d32` hinzufügen. Beispiel:

```
java -d32 ...
```

Quellcode veröffentlichen

Wenn die Anwendungen fertig sind, können Sie in Flash Builder wählen, ob die Benutzer den Quellcode und die Elemente der Anwendung einsehen können. Wie bei HTML können die Benutzer die Quelle im Webbrowser einsehen, indem sie im Kontextmenü „Quelle anzeigen“ auswählen. Das Quellanzeigeprogramm formatiert den Code und setzt ihn farblich ab, damit der Quelltext leicht lesbar ist. Dies ist auch ein praktisches Verfahren, um Code mit anderen Flex- und ActionScript 3.0-Entwicklern auszutauschen.

Option „Quelle anzeigen“ aktivieren

1. Öffnen Sie das fertige Anwendungsprojekt im Editor und wählen Sie „Projekt“ > „Releasebuild exportieren“ aus.

2 Aktivieren Sie „Quellenanzeige aktivieren“ bzw. „Quelle einschließen“ für ActionScript-Projekte.

3 Klicken Sie auf „Quelldateien wählen“.

4 Markieren Sie im Dialogfeld „Anwendungsquelle veröffentlichen“ die Anwendungsdatei bzw. Anwendungsdateien, die in das Menü „Quelle anzeigen“ aufgenommen werden sollen. Standardmäßig ist die Hauptanwendungsdatei markiert.

   Wichtig: *Bei Serverprojekten können Sie beim Exportieren von Quelldateien den Diensteordner festlegen. Das Exportieren von Dateien, die Dienste implementieren, führt zu Auswirkungen auf die Sicherheit. Diese Dateien ermöglichen unter Umständen den Zugriff auf Ihre Datenbank sowie auf Benutzernamen und Kennwörter. Siehe Exportieren von Quelldateien mit der Release-Version einer Anwendung.*

5 (Optional) Ändern Sie den Quellausgabeordner. Standardmäßig wird ein Quellanzeigeordner zum Projektausgabeordner hinzugefügt.

6 Klicken Sie auf „OK“.

Wenn Benutzer die Anwendung ausführen, können sie den Quellcode durch Auswahl von „Quelle anzeigen“ im Kontextmenü einsehen. Der Quellcode erscheint im Standardwebbrowser als Quellstruktur, die der Struktur der Ressourcen (Pakete, Ordner und Dateien), die in der Anwendung enthalten sind und die Sie veröffentlicht haben, entspricht. Wenn ein Quellelement ausgewählt wird, wird der Code im Browser angezeigt. Benutzer können auch den kompletten Satz an Quelldateien herunterladen, indem sie auf den Dateilink „Download.zip“ klicken.

Hinweis: *Aufgrund von Sicherheitsbeschränkungen von Internet Explorer können Sie die Quelle u. U. nicht auf dem lokalen Entwicklungsserver anzeigen. In diesem Fall müssen Sie die Anwendung auf einem Webserver bereitstellen, um die Quelle anzuzeigen.*

Menü „Quelle anzeigen“ zu ActionScript-Projekten hinzufügen

In Flex-Projekten fügen Sie der Anwendung die Option „Quelle anzeigen“ über den Assistenten zum Exportieren von Releasebuilds hinzu. In ActionScript-Anwendungen müssen Sie diese Option manuell hinzufügen.

Das Flex-Framework enthält die folgende Funktion, die Sie in einem Konstruktor einer ActionScript-Anwendung verwenden können, um das Menü „Quelle anzeigen“ zu aktivieren:

```
com.adobe.viewsource.ViewSource.addMenuItem(obj:InteractiveObject, url:String,
hideBuiltins:Boolean = true)
```

Sie können den Code wie folgt in ActionScript-Anwendungen verwenden:

```
package {
    import flash.display.MovieClip;
    import com.adobe.viewsource.ViewSource;

    public class MyASApp extends MovieClip
    {
        public function MyASApp()
        {
            ViewSource.addMenuItem(this, "srcview/index.html");

            // ... additional application code here
        }
    }

}
```

Dieses Beispiel demonstriert, wie das Menü „Quelle anzeigen“ über den Standardpfad des Quellordners (srcview) hinzugefügt wird. Wenn Sie den Pfad des Quellordners ändern, muss der Code den richtigen Pfad verwenden.

Anwendungen ausführen und debuggen

Anwendungen werden basierend auf einer Startkonfiguration ausgeführt (und debuggt). Wenn Sie neue Flex- und ActionScript-Anwendungen erstellen, legt eine Startkonfiguration den Pfad der erstellten Anwendungsdateien und der Hauptanwendungsdatei fest. Sie können die Startkonfiguration ändern oder eine benutzerdefinierte Startkonfiguration erstellen. Weitere Informationen finden Sie unter „Benutzerdefinierte Startkonfigurationen erstellen“ auf Seite 126.

Sie können Projekte auf verschiedene Weisen in Flash Builder ausführen. Sie können zum Beispiel den Befehl „Ausführen“ verwenden, der im Hauptmenü und der Symbolleiste der Workbench, vom Flex Paket-Explorer und über Popupmenüs im Codeeditor verfügbar ist.

Hinweis: *Die Schaltfläche „Ausführen“ hat zwei Elemente: die Hauptaktionsschaltfläche und ein Popupmenü mit den Anwendungsdateien im Projekt, die ausgeführt oder debuggt werden können. Wenn Sie auf die Hauptaktionsschaltfläche klicken, wird die Standardanwendungsdatei ausgeführt. Alternativ können Sie auf das Popupmenü klicken und eine Anwendungsdatei im Projekt auswählen. Anschließend können Sie im Dialogfeld „Konfigurationen erstellen, verwalten und ausführen“ eine Startkonfiguration erstellen oder bearbeiten.*

Startkonfigurationen verwalten

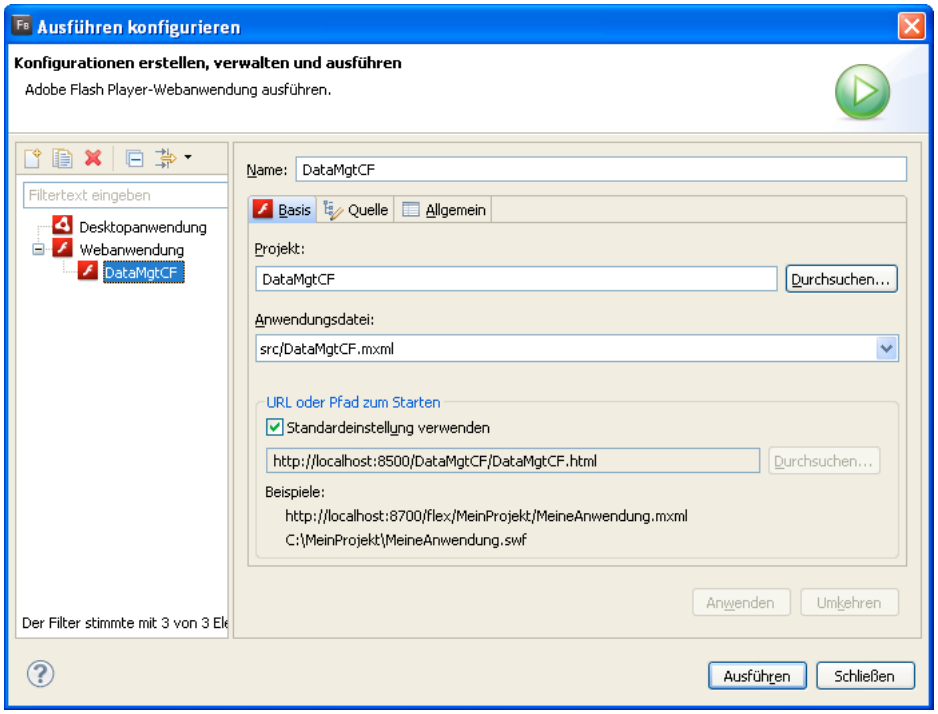
Startkonfigurationen werden sowohl zum Ausführen als auch zum Debuggen von Anwendungen verwendet. Flash Builder enthält eine Standardstartkonfiguration für Flex- und ActionScript-Anwendungen.

Eine Startkonfiguration definiert den Projektnamen, die Hauptanwendungsdatei und den Pfad zur Ausführungs- und Debugversion der Anwendung. Flash Builder enthält eine Standardstartkonfiguration für Anwendungen, mit der Sie für jedes Projekt automatisch Startkonfigurationen erstellen können.

Startkonfigurationen für Webanwendungen erstellen oder bearbeiten

Wenn Sie ein Projekt erstellen, kann es anschließend ausgeführt bzw. debuggt werden. Sowohl die Ausführung als auch das Debuggen der Anwendungen im Projekt wird über eine Startkonfiguration gesteuert. Flash Builder erstellt standardmäßig eine Startkonfiguration für jede Anwendungsdatei im Projekt, wenn Sie die jeweilige Datei zum ersten Mal ausführen oder debuggen. Die Konfigurationen basieren auf der Standardkonfiguration für Anwendungen und können bei Bedarf bearbeitet werden.

Startkonfigurationen werden im Dialogfeld „Konfigurationen erstellen, verwalten und ausführen“ verwaltet.



Startkonfiguration aufrufen und bearbeiten

1 Wählen Sie im Flex Paket-Explorer ein Projekt aus.

2 Öffnen Sie eine Projektdatei im Codeeditor und wählen Sie „Ausführen“ > „Konfigurationen ausführen“.

3 Wählen Sie die Startkonfiguration aus, die bearbeitet werden soll.

4 Ändern Sie die Konfigurationsvoreinstellungen wie gewünscht ab und klicken Sie dann auf „Ausführen“ bzw. „Debuggen“.

Wenn Sie Webprojekte ausführen, wird die SWF-Hauptanwendungsdatei im Standardwebbrowser oder direkt im eigenständigen Flash Player geöffnet.

Verwandte Themen

„Standardwebbrowser ändern“ auf Seite 128

„SWF-Anwendungsdatei im eigenständigen Flash Player ausführen“ auf Seite 128

Startkonfigurationen für Desktopanwendungen erstellen oder bearbeiten

Sie können für Desktop-Anwendungen Startkonfigurationen erstellen, die die von Flash Builder bereitgestellten Standardkonfigurationen überschreiben.

Wenn Sie eine Startkonfiguration für eine Desktopanwendung erstellen oder bearbeiten, geben Sie Befehlszeilenargumente, eine Herausgeber-ID, ein Profil sowie Bildschirmgrößenparameter an. Für Desktop-Anwendungen kann „Profil“ entweder „Desktop“ oder „extendedDesktop“ sein.

Erstellen einer Startkonfiguration:

1 Wählen Sie „Ausführen“ > „Ausführen als“.

Wählen Sie für Debug-Startkonfigurationen „Ausführen“ > „Debugging als“. Wählen Sie für Profil-Startkonfigurationen „Ausführen“ > „Profiling als“.

2 Wählen Sie das Desktop-Projekt. Klicken Sie auf die Schaltfläche „Neue Startkonfiguration“.

Geben Sie die Konfigurationseinstellungen in den Registerkarten „Main“, „Quelle“ oder „Allgemein“ an.

3 Wählen Sie „Ausführen“, „Debuggen“ oder „Profil“, um die Anwendung mit diesen Einstellungen auszuführen.

Klicken Sie auf „Schließen“, um diese Einstellungen zu speichern.

Startkonfigurationen für Mobilanwendungen erstellen oder bearbeiten

Öffnen Sie vor dem ersten Start oder Debuggen einer Mobilanwendung das Dialogfeld „Startkonfigurationen“, um eine Startkonfiguration zu definieren. Für eine Anwendung lassen sich mehrere Startkonfigurationen erstellen. Die Startkonfigurationen, die Sie definieren, werden für das Ausführen und Debuggen der Anwendung eingesetzt.

1 Wählen Sie „Ausführen“ > „Konfigurationen ausführen“, um das Dialogfeld „Konfigurationen ausführen“ zu öffnen.

Um das Dialogfeld „Debugkonfigurationen“ zu öffnen, wählen Sie „Ausführen“ > „Debugkonfigurationen“ (siehe Mobilanwendungen auf einem Gerät ausführen und debuggen.

Sie können auf die beiden Konfigurationsmenüs auch über die Dropdownliste der Schaltfläche „Ausführen“ oder „Debuggen“ in der Symbolleiste von Flash Builder zugreifen.

2 Erweitern Sie den Mobilanwendungsknoten. Klicken Sie in der Symbolleiste des Dialogfelds auf die Schaltfläche „Neue Startkonfiguration“.

3 Wählen Sie eine Zielplattform aus der Dropdownliste.

4 Geben Sie eine Startmethode an:

- Auf Desktop

  Führt die Anwendung auf Ihrem Desktop mit dem AIR Debug Launcher gemäß einer angegebenen Gerätekonfiguration aus. Diese Startmethode ist keine echte Emulation der Ausführung der Anwendung auf einem Gerät. Sie können jedoch das Anwendungslayout anzeigen und mit der Anwendung interagieren.

  Siehe Anwendungsvorschau mit ADL.

  Klicken Sie zum Bearbeiten von Gerätekonfigurationen auf „Konfigurieren“. Siehe Gerätekonfigurationen festlegen.

- Auf Gerät

  Installiert die Anwendung auf Ihrem Gerät und führt sie aus.

  Flash Builder installiert die Anwendung auf Ihrem Gerät und führt sie aus. Flash Builder greift auf das mit dem USB-Port des Computers verbundene Gerät zu. Weitere Informationen finden Sie unter Mobilanwendungen auf einem Gerät ausführen und debuggen.

  Windows-Plattformen benötigen einen USB-Treiber, um ein Android-Gerät mit Ihrem Computer zu verbinden. Weitere Informationen finden Sie unter USB-Gerätetreiber für Android-Geräte (Windows) installieren.

Nach angeschlossenen Geräten suchen

Wenn Sie eine Mobilanwendung auf einem Gerät ausführen oder debuggen, sucht Flash Builder nach angeschlossenen Geräten. Wenn Flash Builder ein einziges angeschlossenes Gerät findet, stellt Flash Builder die Anwendung bereit und startet sie. Ansonsten öffnet Flash Builder das Dialogfenster zum Auswählen eines Geräts für diese Szenarien:

- Kein angeschlossenes Gerät gefunden
- Angeschlossenes Gerät mit Offline-Status gefunden
- Mehrere angeschlossene Geräte gefunden

  Im Dialogfeld „Gerät wählen“ werden die Geräte und deren Status (online oder offline) aufgeführt. Wählen Sie das Gerät aus, das gestartet werden soll.

Mobilanwendungen debuggen

Sie können eine Mobilanwendung von Ihrem Entwicklungsdesktop oder von einem Gerät aus debuggen. Bevor Sie eine Mobilanwendung debuggen können, müssen Sie zunächst eine Startkonfiguration erstellen. Siehe „Startkonfigurationen verwalten“ auf Seite 123.

Als Mindestversion für das Debuggen eines Android-Geräts gilt Android 2.2. Wenn Sie eine Anwendung in Flash Builder auf einem Gerät ausführen, installiert Flash Builder eine Debugversion der Anwendung auf dem Gerät.

Weitere Informationen finden Sie unter Anwendung auf einem Google Android-Gerät debuggen.

Zum Debuggen einer Anwendung auf einem Gerät mit Apple iOS device müssen Sie die Anwendung im IPA-Format speichern und sie manuell auf dem angeschlossenen Gerät installieren.

Weitere Informationen finden Sie unter Anwendung auf einem Apple iOS-Gerät debuggen.

Hinweis: *Wenn Sie einen Releasebuild auf ein Gerät exportieren, installieren Sie eine Nicht-Debugversion der Anwendung. Die Nicht-Debugversion ist jedoch nicht für das Debuggen geeignet.*

Benutzerdefinierte Startkonfigurationen erstellen

Startkonfigurationen unterscheiden sich, je nachdem, ob Sie eine Webanwendung, Desktopanwendung oder Mobilanwendung ausführen. Sie können die Startkonfigurationen, die Flash Builder automatisch für Sie erstellt, anpassen.

1. Wählen Sie im Flash Builder-Menü „Ausführen“ > „Konfigurationen ausführen“.
2. Wählen Sie im Dialogfeld „Konfigurationen erstellen, verwalten und ausführen“ den Anwendungstyp aus, den Sie konfigurieren möchten. Klicken Sie in der Symbolleiste auf die Schaltfläche „Neu“.
3. Geben Sie einen Namen, ein Projekt und eine Anwendungsdatei für die Startkonfiguration an.
4. Ändern Sie nach Bedarf die Konfigurationseigenschaften für den Anwendungstyp, den Sie konfigurieren möchten.
5. Klicken Sie auf „Anwenden“, um die Eigenschaften zu speichern, und auf „Ausführen“, um die Anwendung auszuführen.

Startkonfiguration für das manuelle Starten des Browsers anpassen

Beim Ausführen der Anwendung wird der Browser automatisch von Flash Builder gestartet. Sie können jedoch den Browser manuell starten, anstatt ihn automatisch von Flash Builder starten zu lassen. Gehen Sie dazu wie folgt vor:

1. Wählen Sie aus der Liste der Konfigurationen die Startkonfiguration des Projekts.
2. Deaktivieren Sie auf der Registerkarte „Basis“ die Option „Standardeinstellung verwenden“.
3. Ändern Sie die Standard-URL oder den Standardpfad, die bzw. der gestartet werden soll, in „about:blank“.

Zuletzt gestartete Konfiguration ausführen

❖ Klicken Sie in der Hauptsymbolleiste auf die Schaltfläche „Ausführen“.

Wenn eine Startkonfiguration erstellt werden muss, wird in Flash Builder das Dialogfeld „Startkonfigurationen“ geöffnet.

Verwandte Themen

„Startkonfigurationen für Webanwendungen erstellen oder bearbeiten“ auf Seite 123

„Startkonfigurationen für Desktopanwendungen erstellen oder bearbeiten“ auf Seite 124

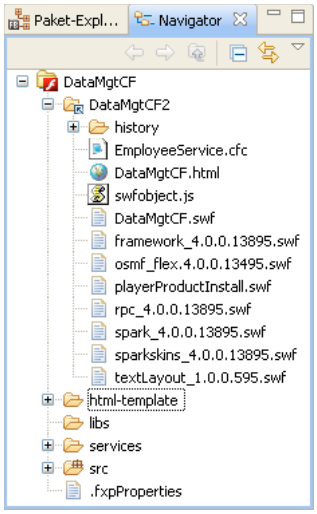
„Startkonfigurationen für Mobilanwendungen erstellen oder bearbeiten“ auf Seite 125

Debugversion der Anwendung

Die Debugversion der Anwendung enthält Debuginformationen, die beim Debuggen der Anwendung verwendet werden. Die exportierte Releasebuildversion enthält nicht die zusätzlichen Debuginformationen und ist daher vom Umfang her kleiner als die Debugversion. Eine HTML-Wrapperdatei enthält einen Link zur SWF-Anwendungsdatei. Die Wrapperdatei wird zum Ausführen oder Debuggen der Anwendung im Webbrowser verwendet.

Hinweis: *Sowohl der Befehl „Ausführen“ als auch der Befehl „Debuggen“ starten den Entwicklungsbuild im Ordner „bin-debug“ (und nicht im exportierten Releasebuildordner „bin-release“).*

In einer typischen Anwendung sieht der Ausgabeordner in etwa wie im folgenden Beispiel gezeigt aus:



Sie können Ihre Flex- und ActionScript-Anwendungen in einem Browser, in AIR oder auf dem Mobilgerät ausführen bzw. debuggen. Sie steuern, wie Anwendungen ausgeführt oder debuggt werden, indem Sie die Startkonfiguration des Projekts ändern (siehe „Startkonfigurationen verwalten“ auf Seite 123).

Verwandte Themen

„Anleitung zum Debuggen einer Anwendung“ auf Seite 141

Debugger-Version von Flash Player verwenden

Standardmäßig wird von Flash Builder die Debuggerversion von Flash Player ausgeführt. Diese Version ist als Browser-Plug-In, ActiveX-Steuerelement und als eigenständige Version verfügbar. Diese Version wird mit Flash Builder installiert. Sie ist jedoch auch als Download von der Adobe-Website verfügbar.

Die Installationsprogramme für die Debuggerversion von Flash Player befinden sich im Ordner „*Flash Builder-Installation*/player“ auf Ihrem Computer.

Mit der Methode `Capabilities.isDebugger()` können Sie programmgesteuert ermitteln, welche Version von Flash Player ausgeführt wird. Weitere Informationen finden Sie unter Determining Flash Player version in Flex.

Wenn Sie in Flash Builder eine Anwendung starten oder debuggen, können Sie die zu verwendende Flash Player-Version angeben.

Verwandte Themen

„Versionsverwaltung für die eigenständige Flash Player-Version“ auf Seite 269

Anwendungen im Browser ausführen und debuggen

Die Startkonfiguration für Webanwendungen legt standardmäßig fest, dass der Ausführungs- und Debugpfad auf die HTML-Wrapper-Dateien im Ausgabeordner des Projekts zeigen. Aus diesem Grund werden Anwendungen in Flash Player, der im Webbrowser läuft, ausgeführt und debuggt. Sie können Anwendungen aber auch im eigenständigen Flash Player ausführen und debuggen (siehe „SWF-Anwendungsdatei im eigenständigen Flash Player ausführen“ auf Seite 128). Sie können auch die Systemeinstellung für den Standardwebbrowser überschreiben und Anwendungen in einem anderen installierten Browser ausführen (siehe „Standardwebbrowser ändern“ auf Seite 128).

Standardwebbrowser ändern

Flash Builder verwendet den Standardwebbrowser zum Ausführen und Debuggen von Anwendungen. Sie können zwar nicht für jede Startkonfiguration einen eigenen Webbrowser festlegen, aber Sie können die Webbrowsereinstellung der Workbench ändern, die bestimmt, wie alle Anwendungen ausgeführt und debuggt werden.

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Webbrowser“ aus.

2 Wählen Sie aus der Liste der Webbrowser, die auf dem System installiert sind, den gewünschten Webbrowser aus.

Hinweis: *Die Option „Internen Webbrowser verwenden“ bezieht sich nicht auf das Ausführen und Debuggen von Anwendungen. Anwendungen werden immer in einem externen Webbrowser ausgeführt und debuggt.*

Sie können auch Browser zur Liste hinzufügen, bearbeiten und aus der Liste entfernen.

3 Klicken Sie auf „OK“, um die Änderungen anzuwenden.

SWF-Anwendungsdatei im eigenständigen Flash Player ausführen

Sie können Anwendungen im eigenständigen Flash Player ausführen und debuggen, indem Sie eine einfache Änderung an der Startkonfiguration vornehmen. Sie können als Programm für das Ausführen und Debuggen der Anwendungen die gewünschte Flash Player-Version angeben. Weitere Informationen hierzu finden Sie unter „Versionsverwaltung für die eigenständige Flash Player-Version“ auf Seite 269.

Anwendungen im eigenständigen Flash Player ausführen und debuggen

1 Wählen Sie im Dialogfeld „Konfigurationen erstellen, verwalten und ausführen“ die Startkonfiguration aus, die Sie ändern möchten.

2 Deaktivieren Sie auf der Registerkarte „Basis“ die Option „Standardwerte verwenden“.

3 Klicken Sie neben dem Pfad für „Ausführen“ bzw. „Debuggen“ auf „Durchsuchen“.

Das Dateiauswahl-Dialogfeld wird aufgerufen und zeigt den Inhalt des Erstellungsausgabeordners an.

4 Wählen Sie die SWF-Anwendungsdatei im Verzeichnis „bin-debug“ aus. Wählen Sie nicht die SWF-Datei im Verzeichnis „bin-release“ aus, sofern dieses Verzeichnis vorhanden ist. Diese SWF-Datei enthält keine Debuginformationen.

5 Klicken Sie auf „Öffnen“, um die Datei auszuwählen. Das Konfigurationsdialogfeld wird erneut angezeigt.

6 Wenden Sie die Änderungen an und führen Sie die Anwendung mit der geänderten Konfiguration aus bzw. debuggen Sie sie hiermit.

Debug- und Nicht-Debugversion der Anwendung vergleichen

Flash Builder generiert standardmäßig Debugversionen der SWF-Datei der Anwendung und speichert diese im Ordner „bin-debug“ des Projekts. Die Anwendung ist größer als die Nicht-Debugversion, weil sie zusätzlichen Code und Metadaten enthält, die der Debugger verwendet.

Wenn Sie eine Nicht-Debugversion der Anwendung generieren möchten, führen Sie einen der folgenden Schritte aus:

- Wählen Sie „Projekt“ > „Releasebuild exportieren“ aus. Hierdurch wird eine Nicht-Debugversion der SWF- oder AIR-Datei im Ordner „bin-release“ erstellt.
- Fügen Sie im Feld „Zusätzliche Compiler-Argumente“ `-debug=false` hinzu. Hierdurch wird immer eine Nicht-Debugversion der SWF-Datei generiert, egal in welchen Ordner Sie die Datei exportieren.

Releaseversion einer Anwendung exportieren

Wenn Sie die Anwendung beendet haben und sie veröffentlichen möchten, können Sie eine Release-Version exportieren. Mit dem Flash Builder-Assistenten „Releasebuild exportieren“ können Sie eine optimierte Version (keine SWF- oder AIR-Debugdatei) der Anwendung in Releasequalität exportieren.

Je nach Anwendungstyp, führt Sie der Assistent durch die einzelnen Schritte, mit denen Sie das Verpacken anpassen und optimieren können.

Nach Ausführen des Assistenten sind weitere Schritte erforderlich, um Ihre Anwendung auf einem Server bereitzustellen.

Webanwendung (wird in Adobe Flash Player ausgeführt)

1 Wählen Sie „Projekt“ > „Releasebuild exportieren“, um den Assistenten „Releasebuild exportieren“ zu öffnen.

2 Wählen Sie das zu konvertierende Projekt und die zu konvertierende Anwendung aus.

3 (Optional) Aktivieren Sie das Kontrollkästchen „Quelle anzeigen aktivieren“, um die Quelldateien der exportierten Anwendung zu veröffentlichen.

Klicken Sie auf „Quelldateien wählen“, um anzugeben, welche Quelldateien in die Auswahl eingeschlossen werden sollen. Zusätzlich zu den angegebenen Quelldateien generiert der Assistent eine ZIP-Archivdatei, die die Quelldateien enthält.

Wichtig: *Bei Serverprojekten können Sie beim Exportieren von Quelldateien den Diensteordner festlegen. Das Exportieren von Dateien, die Dienste implementieren, hat Auswirkungen auf die Sicherheit. Diese Dateien ermöglichen unter Umständen den Zugriff auf Ihre Datenbank sowie auf Benutzernamen und Kennwörter. Siehe Exportieren von Quelldateien mit der Release-Version einer Anwendung.*

4 Klicken Sie auf „Fertig stellen".

5 Kopieren Sie den Ordner, der den exportierten Release enthält, in den Webstamm des Servers, der die Anwendung hostet.

6 (Serverprojekte) Stellen Sie die Dienste und sonstige serverseitige Dateien im Webstamm des Zielservers bereit, wenn Sie einen Releasebuild aus einem Projekt, das einen Anwendungsservertyp angibt, exportieren.

Behalten Sie dieselbe Ordnerstruktur bei, die bei der Entwicklung verwendet wurde.

Dieser Schritt gilt für ColdFusion, PHP, BlazeDS und Data Services. Beim Erstellen eines Projekts in Flash Builder geben Sie den Anwendungsservertyp an.

Wenn sich der Anwendungsserver auf einem anderen Host als die bereitgestellte Anwendung befindet, ist für den Zugriff auf diese Dienste eine Cross-Domain-Richtliniendatei erforderlich. Dies gilt für Projekte, die auf statische XML-Dienstdateien oder lokale Dateien für HTTP-Dienste oder Webdienste zugreifen. Weitere Informationen finden Sie unter Verwenden von domänenübergreifenden Richtliniendateien.

7 (Nur PHP-Serverprojekte) Führen Sie bei PHP-Projekten die folgenden zusätzlichen Schritte aus:

a Installieren Sie das Zend Framework auf dem Server. Siehe Installieren des Zend Framework.

b Modifizieren Sie die Datei `amf-config.ini`, die sich im Ausgabeordner des exportierten Release befindet:

Geben Sie für `zend_path` den absoluten Pfad zum Zend-Installationsordner an.

Setzen Sie `amf.production` auf **true**.

Aktualisieren Sie `webroot` auf den absoluten Pfad zum Webstamm auf dem Server.

Desktopanwendung (ausgeführt in Adobe AIR)

1 (Optional) Ändern Sie die Servereinstellungen in den Projekteigenschaften.

Eine exportierte Desktopanwendung kann ausschließlich auf Dienste zugreifen, die während der Entwicklung verwendet wurden. Wenn Sie den Server für die exportierte Desktopanwendung ändern möchten, modifizieren Sie die Projekteinstellungen.

2 Wählen Sie „Projekt" > „Releasebuild exportieren", um den Assistenten „Releasebuild exportieren" zu öffnen.

3 Wählen Sie das zu konvertierende Projekt und die zu konvertierende Anwendung aus.

Hinweis: *Flash Builder kann keine Anwendungsquelldateien aus exportierten AIR-Anwendungen verfügbar machen.*

4 Geben Sie den Pfad an, in den das Projekt exportiert werden soll. Standardmäßig wird das Projekt in den Projektordner exportiert.

5 Um die AIR-Anwendung zu exportieren, müssen Sie Ihr Build digital signieren, indem Sie einen der folgenden Schritte ausführen:

- Signiertes AIR-Paket exportiert ein digital signiertes AIR-Paket
- Signiertes natives Installationsprogramm erstellt ein digital signiertes Installationsprogramm für die Zielplattform (entweder Windows oder Mac OS X)
- Signierte Anwendung mit gebündelter Laufzeit zum Exportieren der Anwendung mit gebündelter AIR-Laufzeit in der Anwendung.

- „AIRI-Zwischendatei“ erzeugt eine Datei, die später signiert werden kann.

Klicken Sie auf „Weiter“.

Weitere Informationen zum digitalen Signieren finden Sie in der Dokumentation zu Adobe Flash Builder unter „AIR-Anwendungen digital signieren“ auf Seite 136.

6 Auf der Seite „Digitale Signatur“:

Geben Sie das digitale Zertifikat an, das die Identität des Herausgebers der Anwendung zeigt. Um ein selbst signiertes Zertifikat zu generieren, klicken Sie auf „Erstellen“ und geben dann die Daten in die Pflichtfelder ein.

7 Wählen Sie auf der Seite „Inhalt verpacken“ die Ausgabedateien aus, die in die AIR- oder AIRI-Datei aufgenommen werden sollen.

8 Klicken Sie auf „Fertig stellen“.

9 Kopieren Sie das `.air`-Projekt auf den Zieldesktop.

10 (Serverprojekte) Stellen Sie die Dienste auf dem Zielserver bereit, wenn Sie einen Release aus einem Projekt, das einen Anwendungsservertyp angibt, exportieren.

Dieser Schritt gilt für ColdFusion-, PHP-, BlazeDS- und LCDS-Dienste. Beim Erstellen eines Projekts in Flash Builder geben Sie den Anwendungsservertyp an.

11 (Nur PHP-Serverprojekte) Führen Sie bei PHP-Projekten die folgenden zusätzlichen Schritte aus:

a Installieren Sie das Zend Framework auf dem Server. Siehe Installieren des Zend Framework.

b Modifizieren Sie die Datei `amf-config.ini`, die sich im Ausgabeordner des exportierten Release befindet:

Geben Sie für `zend_path` den absoluten Pfad zum Zend-Installationsordner an.

Setzen Sie `amf.production` auf `true`.

Aktualisieren Sie `webroot` auf den absoluten Pfad zum Webstamm auf dem Server.

AIR-Zwischendatei erstellen

Sie können eine AIR-Zwischendatei erstellen, die später signiert werden kann. Verwenden Sie diese Option nur zum Testen.

1 Wählen Sie „Projekt“ > „Releasebuild exportieren“ aus.

Wählen Sie das zu exportierende AIR-Projekt und die Datei, in die das Projekt exportiert werden soll. Klicken Sie auf „Weiter“.

2 Aktivieren Sie die Option „AIRI-Zwischendatei exportieren, die später signiert wird“.

3 (Optional) Klicken Sie auf „Weiter“. Wählen Sie die in die exportierte AIRI-Datei einzuschließenden Ausgabedateien.

Standardmäßig werden alle Dateien mit aufgenommen.

4 Klicken Sie auf „Fertig stellen“.

Nach dem Generieren einer AIR-Zwischendatei kann diese mithilfe des AIR Developer Tool (ADT) signiert werden. Informationen zum ADT-Befehlszeilenprogramm finden Sie in der Adobe AIR-Dokumentation unter Unterzeichnen von AIR-Übergangsdateien mit ADT.

Weitere Informationen zum AIR Developer Tool (ADT) finden Sie in der Adobe AIR-Dokumentation unter www.adobe.com/go/learn_fbairdevelopertool_de.

Anwendung mit Captive Runtime ausführen

Je nachdem, ob Sie das Paket mit einer gebündelten AIR-Laufzeit unter Windows oder Mac exportieren, müssen unterschiedliche Vorgänge durchgeführt werden, um die Anwendung nach der endgültigen Bereitstellung auszuführen.

Unter Windows

❖ Das Anwendungspaket enthält die erforderlichen Anwendungsdateien und die Laufzeit zum Ausführen der Anwendung. Benutzer können die Anwendung sofort nach dem Verpacken ausführen. Außerdem können Sie Tools von Drittanbietern verwenden, um einen Windows Installer (MSI) vom exportierten Anwendungspaket zu erstellen.

Unter Mac

❖ Ziehen Sie zum Ausführen der Anwendung das Anwendungspaket in den Ordner „Anwendungen".

Flex-Projekte in Adobe AIR-Projekte umwandeln

Sie können den Anwendungstyp eines Flex-Projekts von „Web" in „Desktop" ändern. Webanwendungen werden im Flash Player, Desktopanwendungen in Adobe AIR ausgeführt. Bei der Konvertierung werden folgende Änderungen vorgenommen:

- Für jede Anwendung im Projekt wird eine AIR-Deskriptordatei erstellt.
- Die Startkonfigurationen für das Projekt werden aktualisiert, damit sie problemlos in Adobe AIR gestartet werden können.
- Einstellungen für HTML-Wrapper werden entfernt.
- Benutzerdefinierte Flash Player-Einstellungen werden entfernt.
- Der Bibliothekspfad wird so geändert, dass statt der Datei „playerglobal.swc" die Datei „airglobal.swc" enthalten ist.

Bei der Konvertierung können Sie angeben, ob die Application-Tags (Basis-Tags) bei allen Anwendungen des Projekts in WindowedApplication-Tags geändert werden sollen. Wenn Sie angeben, dass diese Tags konvertiert werden sollen, ist dies die einzige Änderung am Anwendungscode, die bei der Konvertierung durchgeführt wird. Überprüfen Sie nach der Konvertierung die Attribute der Basis-Tags, um sicherzustellen, dass die Anwendung wie geplant in Adobe AIR ausgeführt wird.

Webanwendungsprojekte in Desktopanwendungen umwandeln

1 Wählen Sie das zu konvertierende Projekt aus.

   Es sollte ein Flex-Projekt des Anwendungstyps „Web" sein (wird im Flash Player ausgeführt).

2 Wählen Sie im Kontextmenü des Projekts „Projekttyp hinzufügen/ändern" > „In Desktop/Adobe AIR-Projekt konvertieren".

3 Geben Sie im Dialogfeld „In Desktop/Adobe AIR-Projekt konvertieren" an, ob Code umgeschrieben werden soll:

   - Anwendungs-Tags in WindowedApplication konvertieren

     Für vorhandene Anwendungen im Projekt werden alle Application-Tags in WindowedApplication-Tags umgeschrieben. Es werden keine weiteren Änderungen an Ihrem Code vorgenommen. Überprüfen Sie die Attribute der Basis-Tags, um sicherzustellen, dass die Anwendung wie geplant in Adobe AIR ausgeführt wird.

     Neue Anwendungen, die Sie im Projekt erstellen, sind Desktopanwendungen, die in Adobe AIR ausgeführt werden können.

   - Keinen Code umschreiben

Es werden keine Änderungen an Ihrem Code vorgenommen. Sie müssen Anwendungen im Projekt bearbeiten, bevor diese in Adobe AIR ausgeführt werden können.

Neue Anwendungen, die Sie im Projekt erstellen, sind Desktopanwendungen, die in Adobe AIR ausgeführt werden können.

Hinweis: *Dieser Vorgang kann nicht rückgängig gemacht werden.*

Mobilgeräte (ausgeführt in Adobe AIR)

1. Wählen Sie „Projekt“ > „Releasebuild exportieren“, um den Assistenten „Releasebuild exportieren“ zu öffnen.
2. Wählen Sie das Projekt und die Anwendung aus, die Sie exportieren möchten.
3. Wählen Sie die Zielplattformen.
4. Geben Sie den Pfad an, in den das Projekt exportiert werden soll. Sie können das Projekt auf einen Zieldesktop oder ein angeschlossenes Gerät exportieren.
5. Der Basisdateiname ist standardmäßig der Name des Projekts. Sie können den Dateinamen bei Bedarf ändern.
6. Um das Anwendungspaket mit einer digitalen Signatur für jede Zielplattform zu versehen, wählen Sie „Signierte Pakete für jede Zielplattform“.

   Um das Anwendungspaket als digital signierte AIR-Anwendung für den Desktop zu signieren, wählen Sie „Signiertes AIR-Paket für Desktopinstallation“.

   Wenn Sie eine Datei exportieren möchten, die zu einem späteren Zeitpunkt signiert werden soll, exportieren Sie eine AIRI-Zwischendatei.

   Um ein selbst signiertes Zertifikat zu generieren, klicken Sie auf „Erstellen“ und geben Sie dann die Daten in die Pflichtfelder ein.

   Klicken Sie auf „Weiter“.

   Weitere Informationen zum digitalen Signieren finden Sie in der Dokumentation zu Adobe Flash Builder unter „AIR-Anwendungen digital signieren“ auf Seite 136.
7. Auf der Seite „Verpackungseinstellungen“ können Sie das digitale Zertifikat und den Paketinhalt angeben. Die möglichen Einstellungen richten sich nach der Zielplattform. Diese Einstellungen gelten für das Verpacken von Anwendungen mit den Arbeitsabläufen „Ausführen/Debug“ oder „Releasebuild exportieren“.

   Geben Sie auf der Seite „Digitale Signatur“ das digitale Zertifikat P12 an, das die Identität des Anwendungsentwicklers darstellt. Sie können für das Zertifikat auch ein Kennwort angeben.

   Für Apple iOS müssen Sie das Apple iPhone-Zertifikat in das P12-Dateiformat umwandeln und eine Bereitstellungsdatei auswählen. Erwerben Sie zunächst bei Apple ein iPhone-Entwicklerzertifikat und ein Mobile-Provisioning-Profil.

   Weitere Informationen finden Sie unter Erstellung, Debugging oder Bereitstellung einer iOS-Anwendung vorbereiten.

   Dann können Sie das endgültige Releasepaket im Apple App Store oder als Ad-hoc-Verpackung zur begrenzten Verteilung bereitstellen.

   Wählen Sie auf der Seite „Inhalt verpacken“ die Ausgabedateien für die endgültige Version Ihrer Anwendung.

   Hinweis: *Die SWF-Datei und die Anwendungsbeschreibungsdatei dürfen nicht ausgeschlossen werden, da sie für den Export eines gültigen Pakets erforderlich sind.*

Die folgenden Dateien werden nicht benötigt und werden daher automatisch deaktiviert oder ausgeblendet:

- .ipa-Dateien
- Zertifikatsdateien (.p12, .pfx, .provisioning)
- .air- und .airi-Dateien

Für Google Android wählen Sie „Anwendung auf angeschlossenen Geräten installieren und starten“, um das Projekt auf das angeschlossene Gerät zu exportieren. Die exportierte Anwendung wird auf dem Gerät installiert.

Wenn AIR noch nicht auf einem Benutzergerät installiert ist, können Sie eine URL auswählen oder angeben, um AIR für das Anwendungspaket herunterzuladen. Die Standard-URL verweist auf eine Seite im Android App Store. Sie können die Standard-URL jedoch überschreiben und die URL auswählen, die auf eine Seite im Amazon App Store verweist, oder Sie geben Ihre eigene URL ein.

8 Klicken Sie auf „Fertig stellen“.

Weitere Informationen finden Sie unter Verpacken und Exportieren von Mobilanwendungen.

AIR-Anwendungen verpacken

AIR-Anwendungen verpacken

Wenn die Anwendung fertig ist und verteilt (oder auf dem Desktop ausgeführt und getestet) werden kann, verpacken Sie sie in einer AIR-Datei. Das Verpacken besteht aus den folgenden Schritten:

- Auswählen der AIR-Anwendung, die veröffentlicht werden soll
- Optional kann es Benutzern erlaubt werden, den Quellcode anzuzeigen. Anschließend kann ausgewählt werden, welche Anwendungsdateien mit aufgenommen werden sollen
- Auswählen einer generischen AIR-Datei oder eines nativen Installationsprogramms zum Installieren der Anwendung
- Digitales Signieren der AIR-Anwendung mit einem kommerziellen Codesignierungszertifikat oder durch Erstellung und Anwendung einer selbst signierten Signatur
- Optional kann eine AIR-Zwischendatei erstellt werden, die zu einem späteren Zeitpunkt signiert werden kann

AIR-Anwendung verpacken

1 Öffnen Sie das Projekt und vergewissern Sie sich, dass die Anwendung keine Kompilierungsfehler enthält und dass sie wie erwartet läuft.

2 Wählen Sie „Projekt“ > „Releasebuild exportieren“ aus.

3 Wenn Sie mehrere Projekte und Anwendungen in Flash Builder geöffnet haben, wählen Sie das AIR-Projekt aus, das verpackt werden soll.

4 Optional können Sie die Option „Quellenanzeige aktivieren“ aktivieren, wenn Benutzer beim Ausführen der Anwendung den Quellcode sehen dürfen. Sie können einzelne Dateien ausschließen, indem Sie auf „Quelldateien wählen“ klicken. Standardmäßig sind alle Quelldateien ausgewählt. Weitere Informationen zum Veröffentlichen von Quelldateien in Flash Builder finden Sie in der Flash Builder-Hilfe.

Wichtig: *Bei Serverprojekten können Sie beim Exportieren von Quelldateien den Diensteordner festlegen. Das Exportieren von Dateien, die Dienste implementieren, führt zu Auswirkungen auf die Sicherheit. Diese Dateien ermöglichen unter Umständen den Zugriff auf Ihre Datenbank sowie auf Benutzernamen und Kennwörter. Siehe Exportieren von Quelldateien mit der Release-Version einer Anwendung.*

5 Wählen Sie „In AIR-Datei exportieren“ oder „In natives Installationsprogramm exportieren“. Klicken Sie auf „Weiter“.

Mit „In AIR-Datei exportieren“ wird ein generisches Installationsprogramm erstellt, mit dessen Hilfe die Anwendung sowohl auf Windows- als auch auf Mac OS X-Plattformen installiert werden kann.

Mit „In natives Installationsprogramm exportieren“ wird ein jeweils für die Zielplattform (entweder Windows oder Mac OS X) spezifisches Installationsprogramm erstellt.

6 Sie können auch optional den Namen der AIR-Datei ändern, die generiert wird. Klicken Sie anschließend auf „Weiter“, um die Anwendung digital zu signieren.

Projekteigenschaften für verpackte Desktop-Projekte

Sie können das digitale Zertifikat und die Inhalte des Pakets verwalten, indem Sie ein Projekt auswählen und dessen Eigenschaften anzeigen.

1 Wählen Sie im Paket-Explorer ein Projekt aus.

2 Wählen Sie im Hauptmenü „Projekt“ > „Eigenschaften“ oder im Kontextmenü „Eigenschaften“.

3 Wählen Sie im Dialogfeld „Projekteigenschaften“ die Option „Flex-Erstellungsverpackung".

4 Auf der Seite „Digitale Signatur“:

Wählen Sie das digitale Zertifikat aus, das Sie zum Verpacken verwenden möchten, und signieren Sie die AIR-Anwendung. Um ein selbst signiertes Zertifikat zu generieren, klicken Sie auf „Erstellen“ und geben Sie dann die Daten in die erforderlichen Felder ein.

5 Auf der Seite „Inhalt verpacken“:

Heben Sie die Auswahl der Dateien auf, die nicht in der endgültigen Version der Anwendung enthalten sein sollen.

6 Klicken Sie auf „OK“.

Projekteigenschaften für verpackte Mobilprojekte

Sie können das digitale Zertifikat und die Inhalte des Pakets verwalten, indem Sie ein Projekt auswählen und dessen Eigenschaften anzeigen.

1 Wählen Sie im Paket-Explorer ein Projekt aus.

2 Wählen Sie im Hauptmenü „Projekt“ > „Eigenschaften“ oder im Kontextmenü „Eigenschaften“.

3 Wählen Sie im Dialogfeld „Projekteigenschaften“ die Option „Flex-Erstellungsverpackung“ und anschließend die Zielplattform.

4 Auf der Seite „Digitale Signatur“:

Wählen Sie das digitale Zertifikat aus, das Sie zum Verpacken verwenden möchten, und signieren Sie die Mobilanwendung.

Wählen Sie für Apple iOS eine Bereitstellungsdatei aus. Weitere Informationen finden Sie unter Apple iOS-Entwicklungsverfahren mit Flash Builder.

Für Google Android können Sie ein digitales Zertifikat auswählen oder ein selbst signiertes Zertifikat generieren. Um ein selbst signiertes Zertifikat zu generieren, klicken Sie auf „Erstellen“ und geben Sie dann die Daten in die erforderlichen Felder ein.

5 Auf der Seite „Inhalt verpacken“:

Heben Sie die Auswahl der Dateien auf, die nicht in der endgültigen Version der Anwendung enthalten sein sollen.

6 Auf der Seite „Berechtigungen“:

Die Mobilanwendungsberechtigungen für Android können Sie in der Anwendungsbeschreibungsdatei (*app_name*-app.xml) ändern. Weitere Informationen finden Sie unter Mobilanwendungsberechtigungen auswählen.

7 Klicken Sie auf „OK“.

AIR-Anwendungen digital signieren

Beim Exportieren der Release-Version Ihrer Anwendung müssen Sie die AIR-Anwendung digital signieren. Sie haben folgende Möglichkeiten:

- Signieren der Anwendung mithilfe eines handelsüblichen Codesignierungszertifikats
- Erstellen und Verwenden eines selbst signierten digitalen Zertifikats
- Verpacken der Anwendung und spätere Signierung

Digitale Zertifikate, die von Zertifizierungsstellen wie VeriSign, Thawte, GlobalSign und ChosenSecurity ausgegeben wurden, bestätigen den Benutzern Ihre Identität als Herausgeber. Die digitalen Zertifikate belegen außerdem, dass Ihre Installationsdatei unverändert geblieben ist, seit Sie sie signiert haben. Selbst signierte digitale Zertifikate dienen demselben Zweck, sind aber nicht von einem Dritten geprüft.

Sie haben auch die Möglichkeit, die AIR-Anwendung ohne digitale Signatur zu verpacken, indem Sie eine AIR-Zwischendatei (.airi) erzeugen. Eine AIR-Zwischendatei ist nicht gültig, weil sie nicht installiert werden kann. Sie wird stattdessen (vom Entwickler) zum Testen verwendet und kann über das Befehlszeilenprogramm AIR ADT aufgerufen werden. AIR bietet diese Funktion an, weil in einigen Entwicklungsumgebungen ein bestimmter Entwickler bzw. ein bestimmtes Team für das Signieren von Anwendungen verantwortlich ist. Diese Vorgehensweise liefert eine zusätzliche Sicherheitsebene bei der Verwaltung digitaler Zertifikate.

Weitere Informationen zum Signieren von Anwendungen finden Sie in der Adobe AIR-Dokumentation unter Signieren von AIR-Anwendungen.

AIR-Anwendung digital signieren

Sie können die AIR-Anwendung digital signieren, indem Sie ein bestehendes digitales Zertifikat auswählen oder ein neues selbst signiertes Zertifikat erstellen.

1 Wählen Sie „Projekt“ > „Releasebuild exportieren“ aus.

Wählen Sie das zu exportierende AIR-Projekt und die Datei, in die das Projekt exportiert werden soll. Klicken Sie auf „Weiter“.

2 Aktivieren Sie die Option „AIR-Datei exportieren und mit einem digitalen Zertifikat signieren“.

3 Wenn Sie ein digitales Zertifikat haben, klicken Sie auf „Durchsuchen“ und wählen Sie es aus.

4 Um ein neues selbst signiertes digitales Zertifikat zu erstellen, klicken Sie auf „Erstellen“.

5 Geben Sie die erforderlichen Informationen ein und klicken Sie auf „OK“.

6 (Optional) Klicken Sie auf „Weiter“. Wählen Sie die in die exportierte AIRI-Datei einzuschließenden Ausgabedateien.

   Standardmäßig werden alle Dateien mit aufgenommen.

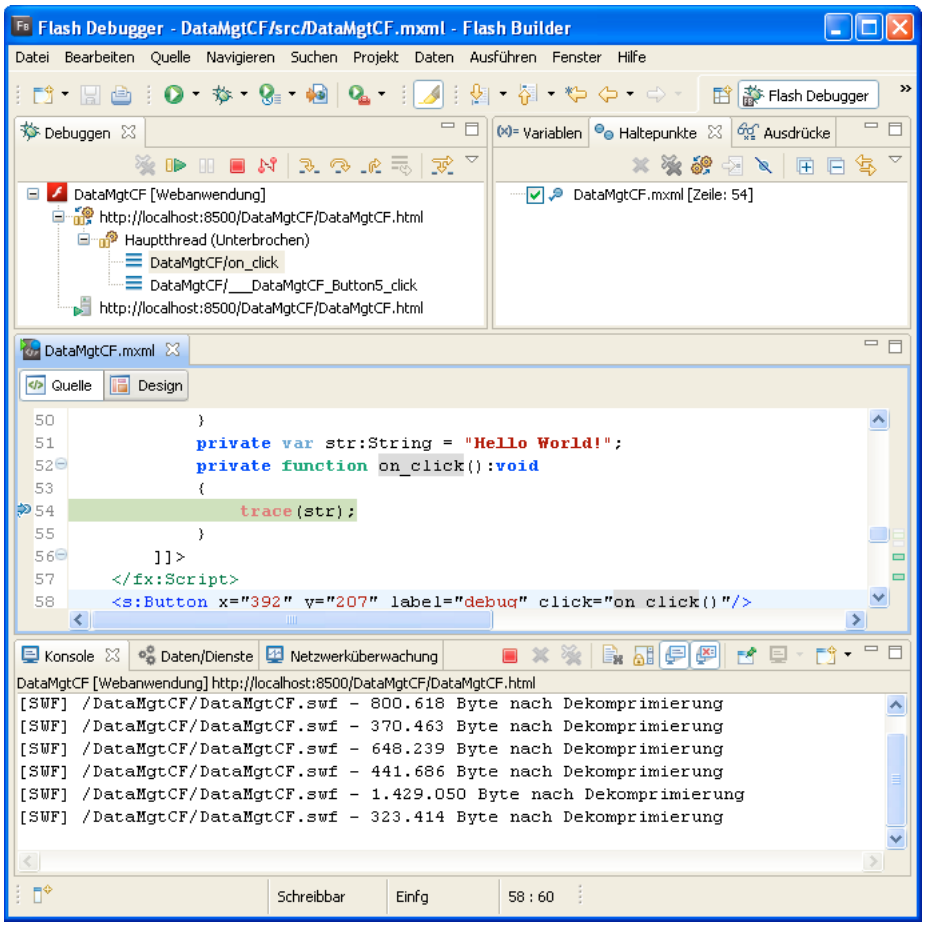
7 Klicken Sie auf „Fertig stellen“, um die AIR-Datei zu generieren.

Kapitel 5: Debugging-Tools in Flash Builder

Flash-Debugperspektive

Die Flash-Debugperspektive enthält einen umfangreichen Debugwerkzeugsatz. Mit den Debugwerkzeugen können Sie Folgendes tun:

- Setzen und Verwalten von Haltepunkten
- Angabe, wie die Anwendung unterbrochen, fortgesetzt und beendet werden soll
- Ausführen bzw. Überspringen bestimmter Codeabschnitte
- Beobachten von Variablen
- Prüfen von Ausdrücken



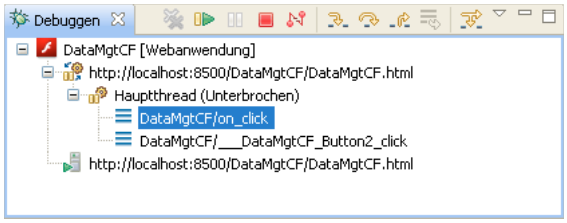
Im Codeeditor können Sie zu ausführbaren Codezeilen Haltepunkte hinzufügen. Die Debugperspektive wird automatisch angezeigt, wenn der erste Haltepunkt erreicht wird. Anschließend wird die Flash-Debugperspektive aktiviert und Sie können den Status der Anwendung mit den Debugwerkzeugen inspizieren bzw. die Anwendung damit verwalten. Weitere Informationen hierzu finden Sie unter „Debug-Sitzungen starten“ auf Seite 142.

Sie können auch manuell zur Debugperspektive wechseln, indem Sie sie in der Perspektivenleiste auswählen. Die Perspektivenleiste befindet sich am rechten Rand der Hauptsymbolleiste der Workbench. Die Debugperspektive enthält die Debug-, Haltepunkte-, Konsolen-, Variablen- und Ausdrückeansicht.

Debugansicht

Die Debugansicht ist die Steuerzentrale der Flash-Debugperspektive. Hier wählen Sie aus, wie die Anwendung ausgeführt, an welchen Stellen sie angehalten, fortgesetzt oder beendet wird und welche Codeabschnitte ausgeführt oder übersprungen werden sollen.

Die Debugansicht (in anderen Debuggern wird diese Ansicht manchmal als *Callstack* bezeichnet) zeigt den Stackframe des angehaltenen Threads der Anwendung an, die Sie gerade debuggen.

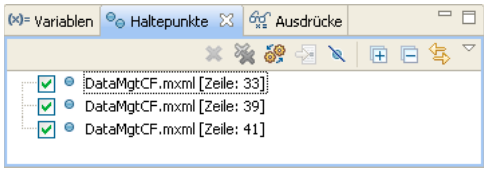


Mit Flex erstellte Anwendungen bestehen aus einem Thread (nicht aus mehreren Threads wie z. B. bei Java). Sie können nur jeweils eine Anwendung debuggen. Wenn Sie eine Anwendung debuggen, sehen Sie daher nur die Prozesse und die Debugansicht für einen einzigen ausgeführten Thread.

Die Debugansicht zeigt eine Liste aller bis zu diesem Punkt aufgerufenen Funktionen in der Reihenfolge ihres Aufrufs an. Die erste aufgerufene Funktion befindet sich zum Beispiel ganz unten in der Liste. Sie können auf eine Funktion doppelklicken, um sie in das Skript zu verschieben. Flash Builder aktualisiert dann die Informationen in der Variablenansicht, um die neue Position im Skript anzuzeigen.

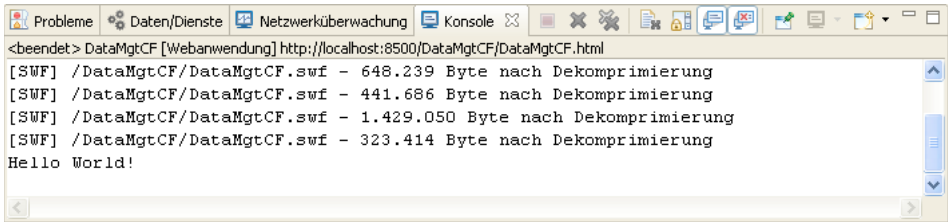
Haltepunkteansicht

Die Haltepunkteansicht listet alle Haltepunkte auf, die Sie im Projekt setzen. Wenn Sie auf einen Haltepunkt doppelklicken, wird seine Position im Editor angezeigt. Sie können auch Haltepunkte deaktivieren, überspringen und entfernen.



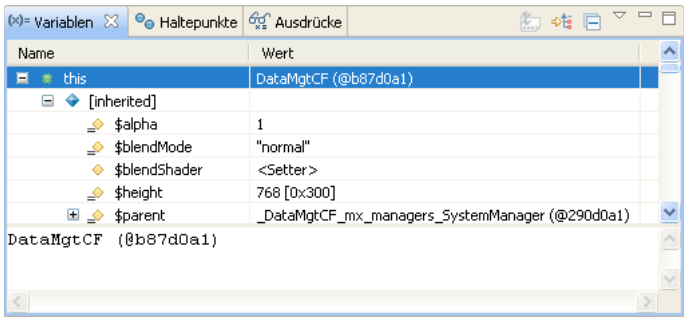
Konsolenansicht

Die Konsolenansicht zeigt die Ausgabe von Trace-Anweisungen an, die im ActionScript-Code platziert sind, und auch Feedback vom Debugger selbst (Status, Warnungen, Fehler usw.).

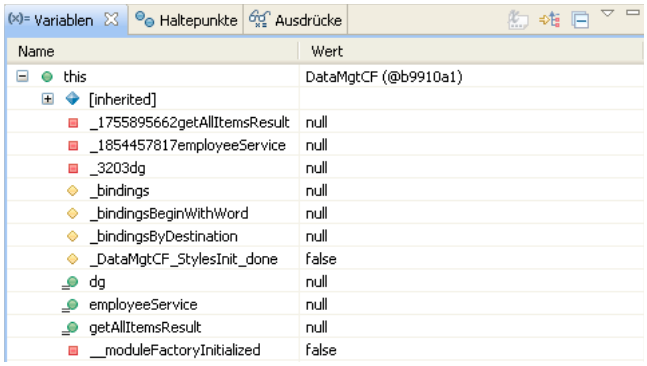


Variablenansicht

Die Variablenansicht zeigt Informationen zu den Variablen in einem ausgewählten Stackframe an. Sie können Variablen auswählen, die (in der Ausdrückeansicht) überwacht werden sollen, und Variablenwerte während der Debug-Sitzung ändern. Während der Debugsitzung können Sie die Änderungen in der momentan ausgeführten SWF-Datei sehen und Lösungen für ein bestehendes Problem testen.



In der Variablenansicht werden Symbole und Overlays zur Darstellung des Variablentyps verwendet.



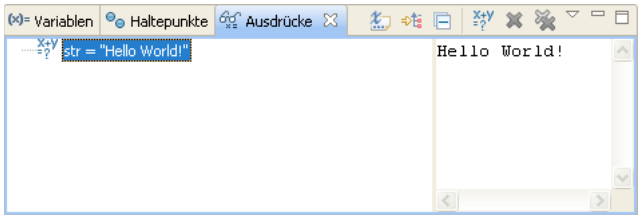
Variablenansicht

Komplexe Variablen können erweitert werden, um ihre Mitglieder anzuzeigen. Sie können die Variablenansicht verwenden, um Variablen hinzuzufügen, indem Sie sie zur Ausdrückeansicht hinzufügen und die Werte der Variablen während der Debug-Sitzung ändern. In der Variablenansicht können Sie auch Beobachtungspunkte setzen (siehe „Überwachungspunkte“ auf Seite 147).

Alle Mitglieder der Superklasse werden in einem separaten Baumknoten gruppiert. Standardmäßig sehen Sie nur die Mitglieder der aktuellen Klasse. Durch diese Gruppierung wird die Zahl der Variablen, die jeweils in der Variablenansicht angezeigt werden, in einem überschaubaren Maß gehalten.

Ausdrückeansicht

Die Ausdrückeansicht wird verwendet, um eine Gruppe kritischer Variablen zu überwachen. Sie können die Variable, die Sie als kritisch erachten, in der Variablenansicht wählen und zur Ausdrückeansicht hinzufügen, um sie zu überwachen. Sie können auch Überwachungsausdrücke hinzufügen und prüfen. Beobachtungsausdrücke sind Codeausdrücke, die bei einer Unterbrechung des Debugvorgangs geprüft werden.



Verwandte Themen

„Anwendungen ausführen und debuggen“ auf Seite 123

„Anleitung zum Debuggen einer Anwendung“ auf Seite 141

Anleitung zum Debuggen einer Anwendung

Nachdem aus Projekten Anwendungen erstellt wurden (siehe „Projekte erstellen“ auf Seite 103), können Sie die Anwendungen in Flash Builder debuggen.

Das Debuggen ähnelt dem Ausführen von Anwendungen. Beim Debuggen bestimmen Sie jedoch, an welchen Stellen im Code die Anwendung angehalten werden soll. Sie können außerdem steuern, ob wichtige Variablen überwacht werden sollen, und Sie können berichtigten Code testen.

Sowohl beim Ausführen als auch beim Debuggen wird eine Konfiguration verwendet, die steuert, wie Anwendungen gestartet werden. Wenn Sie die Anwendung debuggen, führen Sie die Debugversion der Anwendungsdatei aus. Weitere Informationen finden Sie unter „Debugversion der Anwendung“ auf Seite 127.

Sie können die Startkonfiguration bearbeiten, um die Standardhauptanwendungsdatei zu ändern. Sie können auch den Standardstartpfad ändern, um die Anwendung im eigenständigen Flash Player statt in einem Webbrowser auszuführen oder zu debuggen. Weitere Informationen finden Sie unter „SWF-Anwendungsdatei im eigenständigen Flash Player ausführen“ auf Seite 128.

Beim Debuggen von Mobilanwendungen geben Sie in der Startkonfiguration an, ob die Anwendung auf dem Desktop oder auf einem Gerät gestartet werden soll, das an den Computer angeschlossen ist.

Verwandte Themen

„Flash-Debugperspektive“ auf Seite 138

„Anwendungen ausführen und debuggen“ auf Seite 123

„Eclipse-Umgebungsfehler in der Logdatei“ auf Seite 105

Debug-Sitzungen starten

Um eine Debugsitzung zu starten, führen Sie die Anwendungsstartkonfiguration in der Flash-Debugperspektive aus.

Anwendung debuggen

1 Wählen Sie im Flex Paket-Explorer das Projekt aus, das debuggt werden soll.

2 Klicken Sie in der Hauptsymbolleiste der Workbench auf .

Hinweis: *Die Schaltfläche zum Debuggen besteht aus zwei Teilen: der eigentlichen Aktionsschaltfläche und einer Dropdownliste. In der Dropdownliste werden die im Projekt enthaltenen Anwendungsdateien angezeigt, die ausgeführt oder debuggt werden können. Wenn Sie auf die Hauptaktionsschaltfläche klicken, wird die Standardanwendungsdatei des Projekts debuggt. Alternativ können Sie auf die Dropdownliste klicken und eine Anwendungsdatei im Projekt auswählen, die debuggt werden soll. Sie können auch das Dialogfeld „Startkonfiguration" aufrufen und eine Startkonfiguration erstellen oder bearbeiten, indem Sie den Befehl „Debuggen" auswählen.*

Sie können auch „Ausführen" > „Debuggen" auswählen.

Wenn das Projekt noch nicht erstellt wurde, erstellt Adobe® Flash® Builder™ es jetzt und führt es dann im Debugmodus aus.

3 Die Anwendung wird im Standardwebbrowser oder im eigenständigen Flash Player angezeigt und Sie können anschließend über den Flash Builder-Debugger mit ihr arbeiten.

4 Wenn ein Haltepunkt erreicht wird, wird die Flash-Debugperspektive in der Workbench aktiviert.

Debugsitzung in der Plug-In-Konfiguration starten

Der Befehl „Debuggen" funktioniert in der Plug-In-Konfiguration von Flash Builder anders. Statt das aktuell ausgewählte Projekt auszuführen, wird die zuletzt gestartete Konfiguration debuggt. Sie können auch aus einer Liste der zuletzt gestarteten Konfigurationen auswählen.

AIR-Anwendung debuggen

Flash Builder enthält einen kompletten Werkzeugsatz zum Debuggen von AIR-Anwendungen.

1 Öffnen Sie eine Quelldatei für die Anwendung (z. B. eine MXML-Datei) in Flash Builder.

2 Klicken Sie in der Hauptsymbolleiste der Workbench auf .

Die Anwendung wird aufgerufen und in der ADL-Anwendung (dem AIR Debug Launcher) ausgeführt. Der Flash Builder-Debugger entdeckt alle Haltepunkte und Laufzeitfehler und Sie können die Anwendung wie jede andere Flex-Anwendung debuggen.

Mit dem Befehlszeilenprogramm AIR Debug Launcher können Sie Anwendungen auch über die Befehlszeile debuggen. Weitere Informationen finden Sie in der AIR-Dokumentation unter *Verwenden des AIR Debug Launchers (ADL)*.

Haltepunkte setzen und entfernen

Haltepunkte dienen zum Unterbrechen der Ausführung der Anwendung. Auf diese Weise können Sie den Code überprüfen und mithilfe der Debugwerkzeuge von Flash Builder verschiedene Möglichkeiten zur Behebung der Fehler ausprobieren. Beim Debuggen von Anwendungen fügen Sie im Codeeditor Haltepunkte hinzu und verwalten diese dann in der Haltepunkteansicht.

Haltepunkte werden zu ausführbaren Codezeilen hinzugefügt. Der Debugger hält nur an Haltepunkten in Zeilen an, die den folgenden Inhalt haben:

- MXML-Tags, die eine ActionScript-Ereignisprozedur wie z. B. `<mx:Button click="`*dofunction*`()" ...>` enthalten
- ActionScript-Zeilen, die in einem `<mx:Script>`-Tag oder in einer ActionScript-Datei stehen
- Ausführbare Codezeilen in einer ActionScript-Datei

Sie können Haltepunkte beim Schreiben von Code oder beim Debuggen setzen.

Haltepunkt im Codeeditor setzen

1. Öffnen Sie eine Projektdatei, die ActionScript-Code enthält.
2. Suchen Sie die Codezeile, an der Sie einen Haltepunkt setzen möchten, und doppelklicken Sie auf die Markierungsleiste, um einen Haltepunkt hinzuzufügen.

   Die Markierungsleiste befindet sich am linken Rand des Codeeditors.

   Der Haltepunkt wird zur Markierungsleiste und zur Haltepunkteliste in der Haltepunkteansicht der Flash-Debugperspektive hinzugefügt.

Wenn der Debugger auf einen Haltepunkt trifft, wird die Anwendung unterbrochen, die Flash-Debugperspektive wird angezeigt und die Codezeile wird mit einem Haltepunkt markiert. Die Codezeile wird im Codeeditor hervorgehoben. Über die Debugbefehle in der Symbolleiste der Haltepunkteansicht interagieren Sie mit dem Code. Weitere Informationen finden Sie unter „Haltepunkteansicht“ auf Seite 139.

Haltepunkt aus dem Codeeditor entfernen

❖ Doppelklicken Sie in der Markierungsleiste auf einen vorhandenen Haltepunkt.

Der Haltepunkt wird aus der Markierungsleiste und der Haltepunkteansicht der Flash-Debugperspektive entfernt.

Haltepunkte aus der Haltepunkteansicht entfernen

In der Haltepunkteansicht können Sie über die dazugehörige Symbolleiste einen, mehrere oder alle Haltepunkte entfernen..

- Wählen Sie einen oder mehrere Haltepunkte aus der Haltepunkteliste aus und klicken Sie dann auf „Ausgewählte Haltepunkte entfernen“.
- Um alle Haltepunkte auf einmal zu entfernen, klicken Sie auf „Alle Haltepunkte entfernen“.

Sie können auch die Haltepunkte deaktivieren und sie später wieder aktivieren.

Bedingte Haltepunkte setzen

Sie können Bedingungen für Haltepunkte angeben, um den Debugger an Stellen anzuhalten, an denen bestimmte Bedingungen nicht erfüllt sind. Wenn Sie einen bedingten Haltepunkt setzen, geben Sie einen ActionScript-Ausdruck an, der während der Debug-Sitzung überprüft wird. Sie konfigurieren den bedingten Haltepunkt zum Anhalten der Anwendung, wenn eine der folgenden Bedingungen erfüllt ist:

- Der Ausdruck ist „true“.
- Der Wert des Ausdrucks ändert sich.
- Es wird eine bestimmte Trefferanzahl erreicht.

Anleitung zum Setzen eines bedingten Haltepunkts

1 Wählen Sie im Kontextmenü eines Haltepunkts „Haltepunkt-Eigenschaften“.

2 Wählen Sie im Dialogfeld „Haltepunkt-Eigenschaften“ eine oder mehrere der folgenden Optionen aus:

- Aktiviert

  Aktivieren bzw. deaktivieren Sie diese Option, um den Haltepunkt zu aktivieren bzw. zu deaktivieren.

- Trefferanzahl

  Aktivieren Sie die Option „Trefferanzahl“, um einen Zähler für den Haltepunkt zu aktivieren. Geben Sie im Feld „Trefferanzahl“ eine Zahl ein.

  Wenn Sie sowohl „Trefferanzahl“ als auch „Bedingung aktivieren“ aktivieren, bezieht sich die Trefferanzahl darauf, wie oft die angegebene Bedingung erfüllt wurde (die Bedingung ist entweder „true“ oder der Wert der Bedingung ändert sich).

  Wenn Sie nur „Trefferanzahl“ aktivieren, bezieht sich die Trefferanzahl darauf, wie oft der Haltepunkt erreicht wurde.

- Bedingung aktivieren

  Aktivieren Sie die Option „Bedingung aktivieren“ und geben Sie einen ActionScript-Ausdruck ein, der überprüft werden soll. Informationen zu unterstützten Ausdruckstypen finden Sie unter „Beispiele für Ausdrücke“ auf Seite 145.

  Hinweis: *Flash Builder prüft die Syntax des Ausdrucks und meldet Syntaxfehler. Wenn Sie einen Assignment-Operator im Ausdruck haben, zeigt Flash Builder eine Warnung an.*

- Unterbrechen wenn:

  Legen Sie fest, wann die Anwendung angehalten werden soll: wenn der Ausdruck für die Bedingung „true“ ist oder wenn sich der Wert des Ausdrucks ändert.

Variablen in der Variablenansicht verwalten

Wert einer Variablen ändern

1 Wählen Sie die Variable aus, die geändert werden soll.

2 Klicken Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) und wählen Sie im Kontextmenü die Option „Wert ändern“ aus.

3 Geben Sie den neuen Wert ein und klicken Sie auf „OK“.

Die Variable enthält jetzt den neuen Wert.

Geänderte Variablen werden in Rot angezeigt.

Variablen suchen

❖ Um eine Variable oder ein Variablenmitglied in der Variablenansicht zu suchen, wählen Sie die Variablenansicht aus und beginnen Sie, den Namen der Variablen einzugeben, nach der Sie suchen. Sie können auch Platzhalterzeichen (*) verwenden, um nach Wörtern zu suchen, die an einer beliebigen Stelle in einem Variablennamen auftreten (z. B. „*farbe“).

Ausdrückeansicht

Während des Debuggens können Sie den Wert der beobachteten Variable inspizieren und ändern. Sie können auch Beobachtungsausdrücke hinzufügen. Beobachtungsausdrücke sind Codeausdrücke, die bei einer Unterbrechung des Debugvorgangs geprüft werden. Beobachtungsausdrücke eignen sich gut zur Beobachtung von Variablen, die evtl. außerhalb des Bereichs liegen und daher nicht in der Ansicht sichtbar sind, wenn Sie eine andere Funktion aufrufen.

Sie können auch im Quelleditor mit dem Mauszeiger auf einen Ausdruck oder eine Variable zeigen, um den Wert des Ausdrucks bzw. der Variable in einer QuickInfo anzuzeigen. Sie können den Ausdruck zur Ausdrückeansicht hinzufügen, indem Sie mit der rechten Maustaste (unter Mac OS bei gedrückter Ctrl-Taste) klicken und aus dem Menü die Option „Beobachtungsausdruck hinzufügen“ auswählen.

Beispiele für Ausdrücke

Der Flash Builder-Debugger unterstützt eine Vielzahl einfacher und komplexer Ausdrücke. Die folgende Tabelle enthält Beispiele für Ausdrücke, die während einer Debug-Sitzung geprüft werden können. Die Liste ist keineswegs vollständig und soll nur ein paar Beispiele aufzeigen.

Beispiele für unterstützte Ausdrücke

Ausdruck	Beschreibung
myString.length	Gibt die Länge einer Zeichenkette zurück.
myString.indexOf('@')	Verfolgt den Index des @-Zeichens.
"constant string".charAt(0)	Verfolgt das Zeichen an einer bestimmten Position in einer Zeichenkette. Zeichenkettenkonstanten werden unterstützt.
employees.employee.@name	„employees" ist eine XML-Variable. Dieser Ausdruckstyp ist nützlich, wenn E4X-Anwendungen debuggt werden.
x == null	Reservierte Wörter, die für Werte in Ausdrücken stehen.
user1 === user2	Die meisten ActionScript-Operatoren werden unterstützt.
MyClass.myStaticFunc()	Funktionen, die für eine Klasse aufgelöst sind.
this.myMemberFunc()	Funktionen, die für das Schlüsselwort `this` aufgelöst sind.
String.fromCharCode(33)	„String" ist eigentlich eine Funktion (keine Klasse) und „String.fromCharCode" ist ein dynamisches Mitglied dieser Funktion.
myStaticFunc()	Kann nur geprüft werden, wenn „myStaticFunc" von der aktuellen Scope-Chain aus sichtbar ist
myMemberFunc()	Kann nur geprüft werden, wenn „myMemberFunc" von der aktuellen Scope-Chain aus sichtbar ist.
Math.max(1,2,3)	Math-Funktionen werden unterstützt.
mystring.search(/myregex/i)	Reguläre Ausdrücke werden unterstützt.
["my", "literal", "array"]	Erstellung von Arrays.
new MyClass()	Instanziierung von Klassen.
"string" + 3	Handhabt Zeichenkette plus ganze Zahl korrekt.
x >>> 2	Logische Verschiebungsoperatoren werden unterstützt.
3.5 + 2	Führt arithmetische Operationen richtig durch-

Beschränkungen bei der Ausdrucksprüfung

Bei der Ausdrucksprüfung bestehen ein paar Beschränkungen.

- Namespaces werden nicht unterstützt.
- Eingebettete Objekte werden nicht unterstützt.
- Das Schlüsselwort `super` wird nicht unterstützt.
- Vollständig qualifizierte Klassennamen werden nicht unterstützt,

  Sie können zum Beispiel nicht „mx.controls.Button" prüfen.

  Sie können auf den unqualifizierten Klassennamen verweisen. Beispielsweise können Sie definieren, dass „Button" auf „mx.controls.Button" verweist.

  Wenn ein Klassenname zweideutig ist (zwei Klassen mit dem gleichen Namen in verschiedenen Paketen), können Sie nicht steuern, welche Klasse geprüft wird. Sie können aber Folgendes definieren:

  ```
  getDefinitionByName("mx.controls.Button")
  ```

- Die meisten E4X-Ausdrücke können geprüft werden, aber E4X-Filterausdrücke werden nicht unterstützt.

  Sie können zum Beispiel nicht `myxml.(@id=='3'))` prüfen.

- Sie können keine Funktionen aufrufen, die als eine Variable definiert sind.

Überwachungspunkte

Wenn Sie eine Anwendung debuggen, können Sie an bestimmten Variableninstanzen Beobachtungspunkte setzen, um die Anwendung anzuhalten, wenn sich der Wert der beobachteten Variablen ändert. Da Beobachtungspunkte an einer bestimmten Variableninstanz gesetzt werden, können Sie den Beobachtungspunkt nicht im Codeeditor setzen. Setzen Sie Beobachtungspunkte stattdessen über die Variablenansicht, wenn Sie sich in einer Debug-Sitzung befinden.

Beachten Sie beim Setzen von Beobachtungspunkten Folgendes:

- Am Ende einer Debug-Sitzung werden alle Beobachtungspunkte entfernt.
- Sie können keine Beobachtungspunkte auf Gettern setzen, aber Sie können sie im Feld eines Getters setzen.

  Sie können zum Beispiel keinen Beobachtungspunkt an `width` setzen, können aber einen Beobachtungspunkt an `_width` definieren.
- Sie können keine Beobachtungspunkte an lokalen Variablen setzen, können aber Beobachtungspunkte an Mitgliedern lokaler Variablen definieren, wie im folgenden Codefragment gezeigt.

```
public class MyClass
{
   // These are fields of a class, so you can set a watchpoint on
   // 'memberInt', and on 'memberButton', and on 'memberButton._width':
   private var memberInt:int = 0;
   private var memberButton:Button = new Button();

   public function myFunction():void {
      // You CANNOT set a watchpoint on 'i', because it is local:
      var i:int = 0;

      // You CANNOT set a watchpoint on 'someButton', because it is local;
      // but you CAN set a watchpoint on 'someButton._width':
      var someButton:Button = new Button();

...
}
```

- Die Ausführung wird für einen Beobachtungspunkt angehalten, wenn sich der ursprüngliche Wert einer Objektinstanz ändert.

  Dies ist anders als bei einem Ausdruck, der in einem bedingten Haltepunkt verwendet wird, denn in diesem Fall wird die Ausführung in jedem Fall angehalten, wenn sich der Wert einer Variablen ändert.

Überwachungspunkte setzen

❖ In einer Debug-Sitzung können Beobachtungspunkte auf zwei Weisen gesetzt werden:

- Öffnen Sie in der Variablenansicht das Kontextmenü für eine Variable und wählen Sie „Beobachtungspunkt setzen/entfernen“ aus.
- Wählen Sie im Flash Builder-Menü „Ausführen“ die Option „Beobachtungspunkt hinzufügen“ aus.

  Wählen Sie im Dialogfeld „Beobachtungspunkt hinzufügen“ die Variable aus, die beobachtet werden soll.

In der Variablenansicht wird ein Buntstiftsymbol angezeigt. Dieses Symbol bedeutet, dass für die jeweilige Variable ein Beobachtungspunkt gesetzt wurde.

Hinweis: *Wenn Sie versuchen, einen Beobachtungspunkt auf einem Getter zu setzen, öffnet Flash Builder ein Dialogfeld mit Vorschlägen für gültige Variablen für den Beobachtungspunkt. Wenn Sie die vorgeschlagene Variable löschen, werden im Dialogfeld alle gültigen Variablen für das Objekt aufgelistet.*

Funktion „Ausführen bis Zeile“

Flash Builder enthält den Befehl „Ausführen bis Zeile“, um während einer Debugsitzung eine Schleife zu beenden.

Beim Debuggen kann es passieren, dass Ihr Code eine Schleife ausführt, die viele Male wiederholt wird. Mit dem Befehl „Ausführen“ > „Ausführen bis Zeile“ können Sie diese Schleife beenden.

Kapitel 6: Profiling-Tools in Flash Builder

Der Profiler in Adobe Flash® Builder™ hilft Ihnen beim Erkennen von Leistungsengpässen und Speicherlecks in Anwendungen. Der Profiler ist nur in Flash Builder Premium verfügbar.

Flash-Profilingperspektive

Flash Builder Premium bietet eine zusätzliche Perspektive: Flash-Profiling. Die Profiling-Perspektive zeigt mehrere Bedienfelder (oder Ansichten) an, die Profilingdaten unterschiedlich präsentieren. Während Sie die Anwendung bedienen, zeichnet der Profiler Daten zum Status der Anwendung einschließlich der Anzahl der Objekte, der Größe dieser Objekte, der Anzahl der Methodenaufrufe und der in diesen Methodenaufrufen verbrachten Zeit auf.

Profiler-Ansichten

Der Profiler besteht aus mehreren Ansichten (oder Bedienfeldern), die Profilingdaten auf verschiedene Weise präsentieren. Die folgende Tabelle beschreibt die einzelnen Ansichten:

View	**Beschreibung**
Profil	Zeigt die aktuell verbundenen Anwendungen, ihren Status und alle mit ihnen verbundenen Speicher- und Leistungsabbilder an. Zu Anfang weisen die Profilingsitzungen keine Leistungs- oder Speicherabbilder auf.
Gespeicherte Profilingdaten	Zeigt eine Liste gespeicherter Abbilder nach Anwendung geordnet an. Zum Laden gespeicherter Profilingdaten doppelklicken Sie in dieser Liste auf das betreffende Abbild. Weitere Informationen finden Sie unter „Profiling-Daten speichern und laden“ auf Seite 170.
Live-Objekte	Zeigt Informationen zu den von der aktuellen Anwendung verwendeten Klassen an. Diese Ansicht zeigt, welche Klassen instanziiert sind, wie viele erstellt wurden, wie viele sich im Heap befinden und wie viel Speicherplatz die aktiven Objekte belegen. Weitere Informationen finden Sie unter „Informationen in der Live-Objekte-Ansicht anzeigen“ auf Seite 150.
Speicherabbild	Zeigt den Zustand der Anwendung in einem bestimmten Zeitpunkt an. Vergleichen Sie dies mit der Live-Objekte-Ansicht, die laufend den neuesten Stand widerspiegelt. Die Speicherabbildansicht zeigt, auf wie viele Objekte verwiesen wurde, wie viele in der Anwendung eingesetzt wurden und wie viel Speicherplatz jeder Objekttyp zu der betreffenden Zeit beanspruchte. Typischerweise vergleichen Sie zwei Speicherabbilder, die zu verschiedenen Zeiten erstellt wurden, um die Speicherlecks festzustellen. Zeigen Sie die Speicherabbildansicht an, indem Sie auf die Schaltfläche „Speicherabbild erstellen“ klicken und dann in der Profilansicht auf das Abbild doppelklicken. Weitere Informationen finden Sie unter „Speicherabbildansicht“ auf Seite 151.
Verwaiste Objekte	Zeigt die zwischen zwei Speicherabbildern erstellten Objekte an, die sich nach wie vor im Speicher befinden oder nicht mit dem Garbagekollektor aus dem Speicher bereinigt wurden. Sie können auf einen Klassennamen in der Tabelle doppelklicken, um die Objektverweiseansicht zu öffnen. Auf diese Weise können Sie die Beziehungen zwischen den ausgewählten Objekten und den anderen Objekten untersuchen. Zur Anzeige der Verwaiste-Objekte-Ansicht wählen Sie zwei Speicherabbilder und klicken dann auf die Schaltfläche „Verwaiste Objekte“. Weitere Informationen finden Sie unter „Verwaiste-Objekte-Ansicht“ auf Seite 161.

View	Beschreibung
Speicherreservierung	Zeigt Methodenstatistiken beim Vergleichen von zwei Speicherabbildern an. Zur Anzeige der Speicherreservierungsansicht wählen Sie zwei Speicherabbilder aus und klicken dann auf die Schaltfläche „Speicherreservierung einblenden“. Weitere Informationen finden Sie unter „Speicherreservierungsansicht“ auf Seite 155.
Objektverweise	Zeigt Objekte und die Objekte an, die auf diese verweisen. Zur Anzeige der Objektverweiseansicht doppelklicken Sie auf einen Klassennamen in der Speicherabbild- oder der Verwaiste-Objekte-Ansicht. Weitere Informationen finden Sie unter „Objektverweiseansicht“ auf Seite 153.
Objektstatistiken	Zeigt Details zum aufrufenden und aufgerufenen Modul der ausgewählten Objektgruppe an. Zur Anzeige der Objektstatistikenansicht doppelklicken Sie auf einen Eintrag in der Speicherreservierungsansicht. Weitere Informationen finden Sie unter „Objektstatistikansicht“ auf Seite 156.
Leistungsprofil	Zeigt an, wie die Methoden in der Anwendung während eines bestimmten Zeitintervalls leistungsmäßig abgeschlossen haben. Sie klicken dann auf den Namen einer Methode in der Tabelle, um die Methodenstatistikenansicht anzuzeigen und so etwaige Leistungsengpässe zu ermitteln. Zur Anzeige der Leistungsprofilansicht doppelklicken Sie auf eines der Leistungsabbilder in der Profilansicht. Weitere Informationen finden Sie unter „Leistungsprofilansicht“ auf Seite 157.
Methodenstatistiken	Zeigt die Leistungsstatistiken der ausgewählten Methodengruppe an. Zur Anzeige der Methodenstatistikenansicht doppelklicken Sie auf eine Zeile in der Leistungsprofilansicht oder wählen eine Methode im Leistungsprofil aus und klicken auf die Schaltfläche „Methodenstatistiken öffnen“. Weitere Informationen finden Sie unter „Leistungsmerkmale von Methoden erkennen“ auf Seite 159.
Speicherauslastung	Zeigt eine grafische Darstellung der Spitzenauslastung sowie der aktuellen Speicherauslastung über einen gewissen Zeitraum an. Weitere Informationen finden Sie unter „Speicherauslastungsdiagramm“ auf Seite 162.

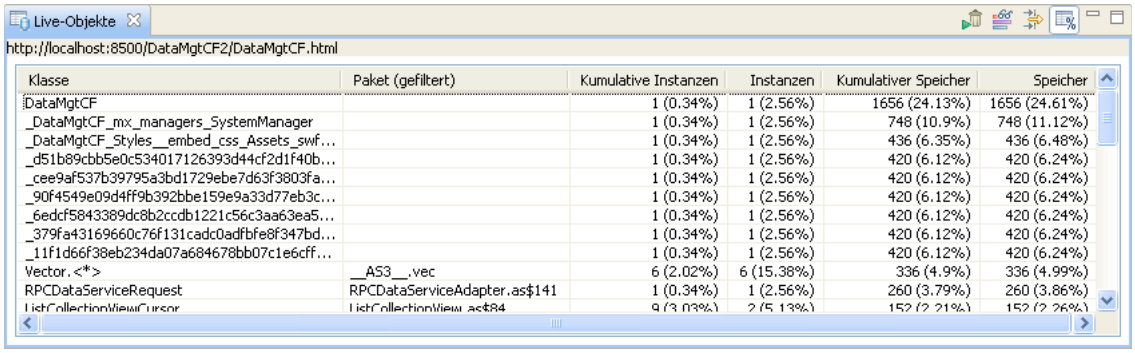
Informationen in der Live-Objekte-Ansicht anzeigen

Die Live-Objekte-Ansicht zeigt Informationen zu den in der aktuellen Anwendung eingesetzten Klassen an. Diese Ansicht zeigt, welche Klassen instanziiert sind, wie viele erstellt wurden, wie viele sich im Speicher befinden und wie viel Speicherplatz die aktiven Objekte belegen.

Der Profiler aktualisiert die Daten in der Live-Objekte-Ansicht laufend, während die Anwendung profiliert wird. Dabei ist es nicht notwendig, die Ansicht zu aktualisieren oder ihr den Fokus zu geben. Die Daten werden auf jeden Fall aktualisiert.

Zur Verwendung der Live-Objekte-Ansicht müssen Sie das Speicher-Profiling beim Starten des Profilers aktivieren. Dies ist die Standardeinstellung. Wenn Sie die Live-Objekte-Ansicht schließen und dann wieder öffnen wollen, öffnen Sie die Dropdownliste in der Profilansicht und wählen „Live-Objekte ansehen“ aus.

Das folgende Beispiel zeigt die Live-Objekte-Ansicht:



Die folgende Tabelle beschreibt die Spalten in der Live-Objekte-Ansicht:

Spalte	**Beschreibung**
Klasse	Die Klassen, die in der aktuell ausgeführten Anwendung Instanzen haben.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Kumulative Instanzen	Die Gesamtzahl der Instanzen jeder Klasse, die seit dem Starten der Anwendung erstellt wurden.
Instanzen	Die Anzahl der Instanzen jeder Klasse, die derzeit im Speicher sind. Dieser Wert ist immer kleiner oder gleich dem Wert in der Spalte „Kumulative Instanzen“.
Kumulativer Speicher	Die Gesamtmenge an Speicher (in Byte), die von allen Instanzen jeder Klasse belegt wird, einschließlich der nicht mehr im Speicher befindlichen Klassen.
Speicher	Die Gesamtmenge an Speicher (in Byte), die von allen Instanzen jeder Klasse derzeit belegt wird. Dieser Wert ist immer kleiner oder gleich dem Wert in der Spalte „Kumulativer Speicher“.

In der Regel verwenden Sie die Daten in der Live-Objekte-Ansicht, um zu sehen, wie viel Speicher von den Objekten belegt wird. Wenn Objekte vom Garbagekollektor aus dem Speicher bereinigt werden, reduziert sich die Anzahl der Instanzen und die Menge des belegten Speichers, während sich bei den kumulativen Instanzen und der kumulativen Speicherauslastung eine Erhöhung abzeichnet. Dieser Ansicht können Sie auch entnehmen, wie viel Speicher beansprucht wird, während die Anwendung läuft.

Weitere Informationen zum Ausführen und Auswerten der Ergebnisse der Speicherbereinigung (Garbage Collection) finden Sie unter „Garbage Collection“ auf Seite 172.

Zur Einschränkung der Daten in der Live-Objekte-Ansicht können Sie die Profilerfilter verwenden. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.

Speicherabbildansicht

Die Speicherabbildansicht zeigt Informationen zu den Objekten und der Speicherauslastung einer Anwendung in einem ganz bestimmten Zeitpunkt an. Anders als die Live-Objekte-Ansicht wird sie nicht laufend aktualisiert.

Um die Speicherabbildansicht verwenden zu können, müssen Sie beim Starten des Profilers das Speicher-Profiling aktivieren. Dies ist die Standardeinstellung.

Speicherabbild erstellen und anzeigen

1 Starten Sie eine Profilingsitzung.

2 Interagieren Sie mit der Anwendung, bis Sie einen Punkt im Status der Anwendung erreichen, von dem Sie ein Speicherabbild erstellen möchten.

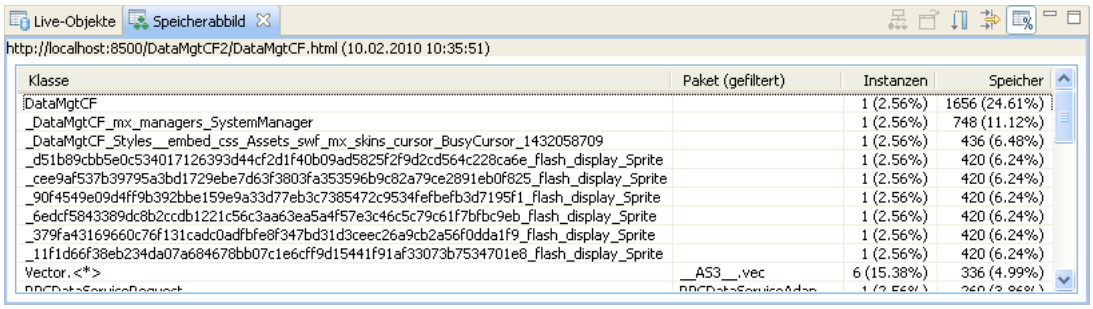
3 Wählen Sie die Anwendung in der Profilansicht aus.

4 Klicken Sie auf die Schaltfläche „Speicherabbild erstellen“.

Der Profiler erstellt ein Speicherabbild und versieht es mit einem Zeitstempel. Außerdem löst er automatisch eine Speicherbereinigung aus, bevor das Speicherabbild aufgezeichnet wird.

5 Zur Anzeige der Daten im Speicherabbild doppelklicken Sie in der Profilansicht auf das Abbild.

Das folgende Beispiel zeigt die Speicherabbildansicht:



Die folgende Tabelle beschreibt die Spalten in der Speicherabbildansicht:

Spalte	Beschreibung
Klasse	Die Klassen, die im Zeitpunkt der Erstellung des Speicherabbilds Instanzen im Speicher hatten.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Instanzen	Die Anzahl der im Speicher befindlichen Instanzen jeder Klasse im Zeitpunkt der Erstellung des Speicherabbilds.
Speicher	Die Menge Speicher (in Byte), die alle Instanzen jeder Klasse im Zeitpunkt belegten, in dem das Speicherabbild aufgezeichnet wurde.

In der Regel verwenden Sie ein Speicherabbild als Ausgangspunkt, um zu bestimmen, auf welche Klassen Sie sich bei der Speicheroptimierung konzentrieren sollten, oder um Speicherlecks aufzufinden. Zu diesem Zweck erstellen Sie mehrere Speicherabbilder zu verschiedenen Zeitpunkten und ermitteln dann die Unterschiede in den Ansichten „Verwaiste Objekte“ und „Speicherreservierung“.

Sie können Speicherabbilder auch speichern, um den Status einer Anwendung in verschiedenen Profilingsitzungen zu vergleichen. Weitere Informationen finden Sie unter „Profiling-Daten speichern und laden“ auf Seite 170.

Wenn Sie in der Speicherabbildansicht auf eine Zeile doppelklicken, zeigt der Profiler die Objektverweiseansicht an. Diese Ansicht zeigt die Stacktraces für die Instanzen der aktuellen Klasse an. Die Stacktraces der Instanzen der aktuellen Klasse sehen Sie in der Objektverweiseansicht. Weitere Informationen zu dieser Ansicht finden Sie unter „Objektverweiseansicht“ auf Seite 153.

Über Profilerfilter können Sie auch angeben, welche Daten in der Speicherabbildansicht angezeigt werden sollen. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.

Objektverweiseansicht

Die Objektverweiseansicht zeigt Stacktraces für Klassen an, die in der Anwendung instanziiert wurden.

In der Ansicht „Objektverweise“ werden Informationen zu den Instanzen der gewählten Klassen in einer Baumstruktur angezeigt. Die Ansicht „Objektverweise“ zeigt auch Pfade zu den Rückverweisen des Objekts an, die zum GC-Stamm zurückführen. Wenn mehr als eine Instanz eines Pfades zu einem Objektverweis vorhanden ist, wird der Pfad nicht wiederholt.

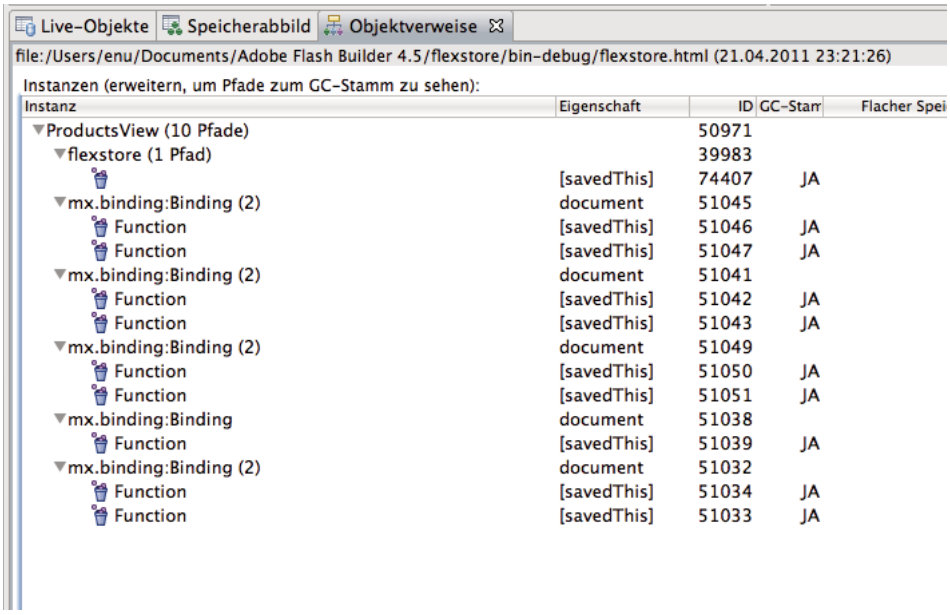
Zum Öffnen der Ansicht „Objektverweise“ doppelklicken Sie in der Speicherabbild- oder der Verwaiste-Objekte-Ansicht auf einen Klassennamen.

Die Objektverweiseansicht zeigt Daten in zwei Tabellen (in der Instanzen- und der Speicherreservierungstabelle) an.

Die Instanzentabelle enthält alle Objekte mit Verweisen auf das aktuelle Objekt. Die Zahl in Klammern nach den Klassennamen wird in der Strukturansicht als Knoten dargestellt. Die Zahl gibt die Anzahl der Pfade an, die von diesem Knoten zum GC-Stamm führen.

Es ist nicht möglich, die Anzahl der Vorwärtsverweise für ein Objekt anzuzeigen. Objekte ohne Verweise werden in dieser Tabelle nicht aufgeführt. Dies ist jedoch kaum möglich, denn ein Objekt ohne Verweise hätte vom Garbagekollektor aus dem Speicher bereinigt werden müssen.

Das folgende Beispiel zeigt die Instanzentabelle in der Objektverweiseansicht:



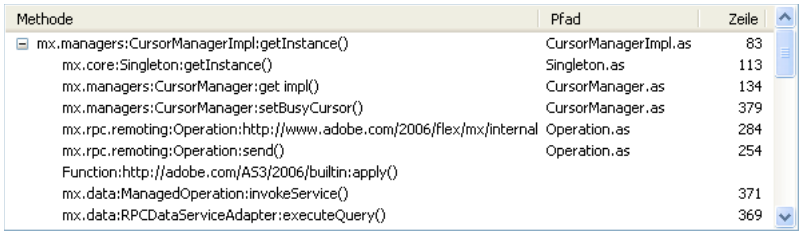
Die folgende Tabelle beschreibt die Spalten in der Instanzentabelle:

Spalte	Beschreibung
Instanz	Die Klasse des Objekts, das einen Verweis auf das angegebene Objekt enthält. Erweitern Sie eine Instanz einer Klasse, um die Pfade zum Objekt anzuzeigen. Die Anzahl der Pfade, die angezeigt werden, ist in der Speicherabbildansicht über die Filteroption konfigurierbar.
Eigenschaft	Die Eigenschaft des Objekts, das einen Verweis auf das angegebene Objekt enthält. Angenommen, Sie haben ein Objekt „o“ mit einer Eigenschaft `i` und diese Eigenschaft zeigt auf die Schaltflächenbeschriftung: `o.i = myButton.label;` In diesem Fall wird ein Verweis auf `myButton.label` von der Eigenschaft `i` erstellt.
ID	Die Verweis-ID des Objekts, das den Verweis auf das ausgewählte Objekt enthält.
GC-Stamm	Gibt an, ob ein Objekt einen Rückverweis zum GC-Stamm hat. Um zu sehen, ob ein Rückverweis zu einem GC-Stamm vorhanden ist, erweitern Sie einen Objektknoten in der Strukturansicht.
Flacher Speicher	Gibt die Größe des Objekts an.
Beibehaltener Speicher	Gibt die Größe des Objekts und die Größe aller anderen Objekte an, auf die es verweist.

Die Eigenschaften eines Objekts werden in der Ansicht „Objekteigenschaften“ angezeigt. Doppelklicken Sie auf eine Eigenschaft, um die Eigenschaftshierarchie dieses Objekts anzuzeigen.

Die Speicherreservierungstabelle zeigt das Stacktrace für die ausgewählte Instanz in der Instanzentabelle an. Wenn Sie in der Instanzentabelle eine Instanz wählen, zeigt der Profiler den Call-Stack für diese Instanz in der Speicherreservierungstabelle an.

Das folgende Beispiel zeigt die Speicherreservierungstabelle in der Objektverweiseansicht:



Die folgende Tabelle beschreibt die Spalten in der Speicherreservierungstabelle:

Spalte	Beschreibung
Methode	Die oberste Methode in dieser Tabelle ist die Methode, die die in der Instanzentabelle aufgeführte Instanz der Klasse erstellt hat. Sie können die Methode erweitern, um das Stacktrace der Methode anzuzeigen. Dies kann nützlich sein, um zu ermitteln, wo der Call-Stack beginnt.
Pfad	Die Datei, in der die Methode definiert ist.
Zeile	Die Zeilennummer in der Datei.

Daten in dieser Tabelle können Sie nur anzeigen, wenn Sie beim Starten des Profilers die Speicherreservierungen aktivieren.

Zum Öffnen des Quellcodes der ausgewählten Klasse doppelklicken Sie in dieser Tabelle auf eine Klasse.

Speicherreservierungsansicht

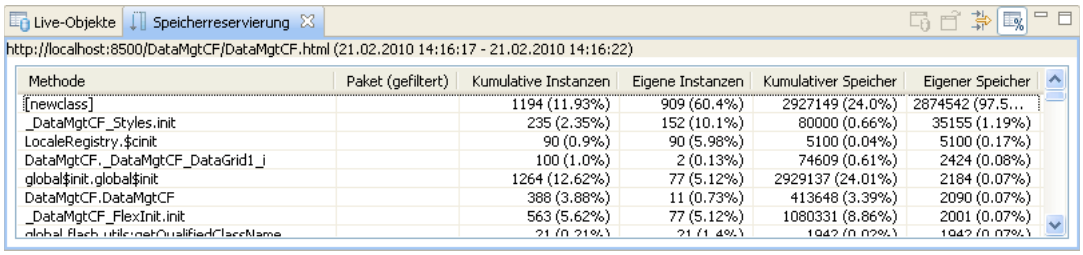
Die Speicherreservierungsansicht zeigt, welche Methoden zwischen zwei Speicherabbildern aufgerufen wurden und wie viel Speicherplatz während dieser Aufrufe beansprucht wurde. Zum Öffnen der Speicherreservierungsansicht wählen Sie zwei Speicherabbilder aus und klicken dann auf die Schaltfläche „Speicherreservierung einblenden“. Weitere Informationen zur Aufzeichnung eines Speicherabbilds finden Sie unter „Speicherabbildansicht“ auf Seite 151.

Das Ergebnis des Vergleichs zwischen den Speicherabbildern ist eine Liste mit Methoden, die der Flash Player zwischen den beiden Abbildern ausführte. Für jede dieser Methoden meldet der Profiler die Anzahl der in dieser Methode erstellten Objekte.

Sie können diese Informationen zur Optimierung der Leistung heranziehen. Nachdem Sie die Methoden identifiziert haben, die eine übermäßige Anzahl von Objekten erstellen, können Sie diese Problembereiche optimieren.

Um die Speicherreservierungsansicht verwenden zu können, müssen Sie beim Starten des Profilers die Speicherreservierungen aktivieren. Standardmäßig ist diese Einstellung deaktiviert.

Das folgende Beispiel zeigt die Speicherreservierungsansicht:


Live-Objekte
Speicherreservierung
http://localhost:8500/DataMgtCF/DataMgtCF.html (21.02.2010 14:16:17 - 21.02.2010 14:16:22)
Methode | Paket (gefiltert) | Kumulative Instanzen | Eigene Instanzen | Kumulativer Speicher | Eigener Speicher
[newclass] | 1194 (11.93%) | 909 (60.4%) | 2927149 (24.0%) | 2874542 (97.5...
_DataMgtCF_Styles.init | 235 (2.35%) | 152 (10.1%) | 80000 (0.66%) | 35155 (1.19%)
LocaleRegistry.$cinit | 90 (0.9%) | 90 (5.98%) | 5100 (0.04%) | 5100 (0.17%)
DataMgtCF._DataMgtCF_DataGrid1_i | 100 (1.0%) | 2 (0.13%) | 74609 (0.61%) | 2424 (0.08%)
global$init.global$init | 1264 (12.62%) | 77 (5.12%) | 2929137 (24.01%) | 2184 (0.07%)
DataMgtCF.DataMgtCF | 388 (3.88%) | 11 (0.73%) | 413648 (3.39%) | 2090 (0.07%)
_DataMgtCF_FlexInit.init | 563 (5.62%) | 77 (5.12%) | 1080331 (8.86%) | 2001 (0.07%)

Die folgende Tabelle beschreibt die Spalten in der Speicherreservierungsansicht:

Spalte	**Beschreibung**
Methode	Die während des Speicherabbildintervalls aufgerufene Methode. Diese Spalte enthält auch die Klasse, deren Instanz diese Methode aufrief.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Kumulative Instanzen	Die Anzahl der in dieser Methode instanziierten Objekte sowie alle von dieser Methode aufgerufenen Methoden.
Eigene Instanzen	Die Anzahl der in dieser Methode instanziierten Objekte. Dies schließt keine Objekte ein, die in nachträglichen Methodenaufrufen instanziiert wurden.
Kumulativer Speicher	Die Menge Speicher (in Byte), die von den in dieser Methode instanziierten Objekten sowie allen von dieser Methode aufgerufenen Methoden beansprucht wurde.
Eigener Speicher	Die Menge Speicher (in Byte), die von den in dieser Methode instanziierten Objekten beansprucht wird. Dies schließt nicht den Speicher ein, der von Objekten belegt ist, die in nachfolgenden Methodenaufrufen dieser Methode instanziiert werden.

Beim Aufzeichnen von Methoden während Sampling-Intervallen zeichnet der Profiler auch interne Flash Player-Aktionen auf. Diese Aktionen werden in der Methodenliste in Klammern angezeigt und erscheinen als `[mouseEvent]`, `[newclass]` bzw. mit ähnlichen Namen. Weitere Informationen zu internen Flash Player-Aktionen finden Sie unter „Funktionsweise des Profilers“ auf Seite 164.

Zum Öffnen der Objektstatistikenansicht klicken Sie auf eine Zeile in der Speicherreservierungstabelle. Diese Ansicht zeigt Details zu den Objekten an, die in der ausgewählten Methode erstellt wurden. Außerdem können Sie hier Details zu Objekten anzeigen, die in Methoden erstellt wurden, die von dieser Methode aufgerufen wurden. Weitere Informationen finden Sie unter „Objektstatistikansicht“ auf Seite 156.

Zur Einschränkung der Daten in der Speicherreservierungsansicht verwenden Sie die Profilerfilter. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.

Objektstatistikansicht

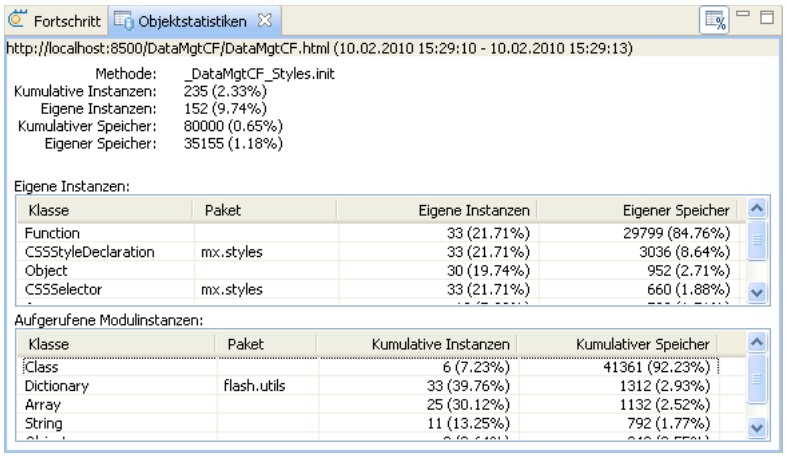
Diese Ansicht zeigt die Leistungsstatistiken der ausgewählten Objektgruppe an. Anhand dieser Informationen können Sie bestimmen, welche Methodenaufrufe eine unverhältnismäßig große Zahl anderer Methoden aufrufen. Außerdem sehen Sie, wie viel Speicher die in diesen Methoden instanziierten Objekte beanspruchen. Die Objektstatistikenansicht wird zur Ermittlung potenzieller Speicherlecks und anderer Leistungsprobleme in der Anwendung verwendet.

Für den Zugriff auf die Objektstatistikenansicht wählen Sie in der Profilansicht zwei Speicherabbilder aus und sehen sich den Vergleich in der Speicherreservierungsansicht an. Anschließend doppelklicken Sie auf eine Zeile, um die Details in der Objektstatistikenansicht anzuzeigen.

Die Ansicht enthält drei Bereiche:

- Eine Zusammenfassung der Statistiken des ausgewählten Objekts, einschließlich der Anzahl der Instanzen und der Menge des belegten Speichers.
- Eigene Instanzen-Tabelle: Eine Liste der Objekte, die in der in der Speicherreservierungsansicht ausgewählten Methode instanziiert wurden. Dies schließt keine Objekte ein, die in nachträglichen Methodenaufrufen instanziiert wurden. Die Anzahl der Objekte in dieser Liste entspricht der Anzahl der Objekte, die in der Spalte „Eigene Instanzen“ in der Speicherreservierungsansicht angegeben sind.
- Aufgerufene Instanzen-Tabelle: Eine Liste der Objekte, die in Methoden instanziiert wurden, die durch die in der Speicherreservierungsansicht ausgewählten Methoden aufgerufen wurden. Dies schließt keine Objekte ein, die selbst in der Methode instanziiert wurden. Die Anzahl der Objekte in dieser Liste entspricht der Anzahl der Objekte in der Spalte „Kumulative Instanzen“ in der Speicherreservierungsansicht.

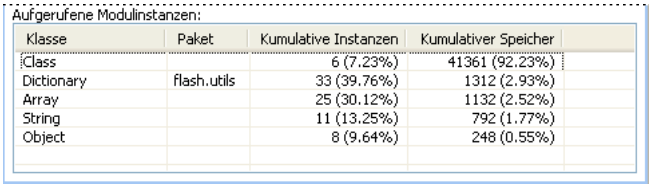
Das folgende Beispiel zeigt die Methodenzusammenfassung und die Tabellen mit den eigenen und den aufgerufenen Instanzen der Objektstatistikenansicht:


Fortschritt
Objektstatistiken
http://localhost:8500/DataMgtCF/DataMgtCF.html (10.02.2010 15:29:10 - 10.02.2010 15:29:13)
Methode: _DataMgtCF_Styles.init
Kumulative Instanzen: 235 (2.33%)
Eigene Instanzen: 152 (9.74%)
Kumulativer Speicher: 80000 (0.65%)
Eigener Speicher: 35155 (1.18%)
Eigene Instanzen:
Klasse Paket Eigene Instanzen Eigener Speicher
Function 33 (21.71%) 29799 (84.76%)
CSSStyleDeclaration mx.styles 33 (21.71%) 3036 (8.64%)
Object 30 (19.74%) 952 (2.71%)
CSSSelector mx.styles 33 (21.71%) 660 (1.88%)
Aufgerufene Modulinstanzen:
Klasse Paket Kumulative Instanzen Kumulativer Speicher
Class 6 (7.23%) 41361 (92.23%)
Dictionary flash.utils 33 (39.76%) 1312 (2.93%)
Array 25 (30.12%) 1132 (2.52%)
String 11 (13.25%) 792 (1.77%)

Die folgende Tabelle beschreibt die Felder in der eigenen Instanzen-Tabelle in der Objektstatistikenansicht:

Spalte	Beschreibung
Klasse	Die Klassen, die nur in der ausgewählten Methode instanziiert wurden. Dies schließt keine Klassen ein, die in nachfolgenden Aufrufen aus dieser Methode instanziiert wurden.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket" leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Eigene Instanzen	Die Anzahl Instanzen dieser Klasse, die ausschließlich in der ausgewählten Methode erstellt wurden. Dies schließt keine Instanzen ein, die in nachfolgenden Aufrufen von dieser Methode erstellt wurden.
Eigener Speicher	Die Menge Speicher (in Byte), die von Instanzen belegt wird, die nur in der ausgewählten Methode erstellt wurden. Dies schließt nicht den Speicherplatz ein, der von Instanzen belegt wird, die in nachfolgenden Aufrufen von dieser Methode erstellt wurden.

Das folgende Beispiel zeigt die aufgerufene Instanzen-Tabelle der Objektstatistikenansicht:


Aufgerufene Modulinstanzen:

Klasse	Paket	Kumulative Instanzen	Kumulativer Speicher
Class		6 (7.23%)	41361 (92.23%)
Dictionary	flash.utils	33 (39.76%)	1312 (2.93%)
Array		25 (30.12%)	1132 (2.52%)
String		11 (13.25%)	792 (1.77%)
Object		8 (9.64%)	248 (0.55%)

Die folgende Tabelle beschreibt die Felder in der aufgerufenen Instanzen-Tabelle der Objektstatistikenansicht:

Spalte	Beschreibung
Klasse	Die Klassen, die in der ausgewählten Methode instanziiert wurden. Dies schließt Klassen ein, die in nachfolgenden Aufrufen von dieser Methode instanziiert wurden.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket" leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Kumulative Instanzen	Die Anzahl Instanzen dieser Klasse, die in der ausgewählten Methode sowie in nachfolgenden Aufrufen von dieser Methode erstellt wurden.
Kumulativer Speicher	Die Menge Speicher (in Byte), die von Instanzen belegt wird, die in der ausgewählten Methode und in nachfolgenden Aufrufen von dieser Methode erstellt wurden.

Leistungsprofilansicht

Die Leistungsprofilansicht ist beim Leistungs-Profiling als die primäre Ansicht zu verwenden. Sie zeigt Statistiken wie die Anzahl der Aufrufe, die eigene Zeit und die kumulative Zeit für die Methoden an, die während eines bestimmten Sampling-Intervalls aufgerufen werden. Diese Daten dienen zur Erkennung von Leistungsengpässen.

Beim Leistungs-Profiling wird eine Liste mit Methoden, zusammen mit Informationen zu diesen Methoden, die im Zeitraum zwischen dem Löschen der Leistungsdaten und der Erfassung neuer Daten aufgerufen werden, gespeichert. Dieser Zeitraum wird als *Interaktionsperiode* bezeichnet.

Um die Leistungsprofilansicht zu verwenden, müssen Sie beim Starten des Profilers das Leistungs-Profiling aktivieren. Dies ist die Standardeinstellung.

Leistungsprofil generieren

1 Beginnen Sie eine Profilingsitzung mit aktiviertem Leistungs-Profiling.

2 Interagieren Sie so lange mit der Anwendung, bis Sie den Punkt erreichen, an dem Sie mit dem Profiling beginnen möchten.

3 Klicken Sie auf die Schaltfläche „Leistungsdaten zurücksetzen“.

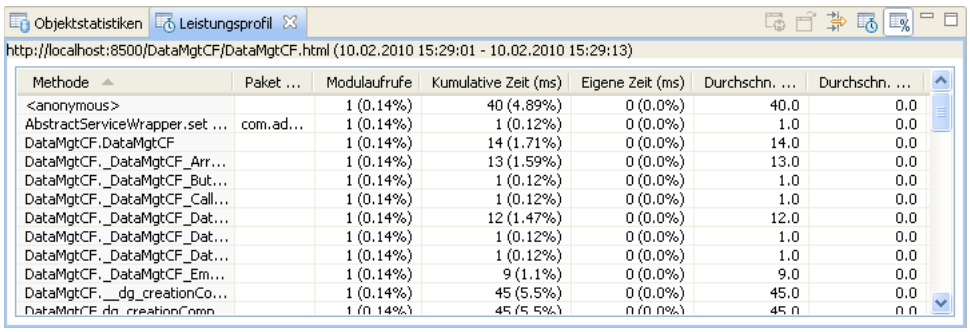
4 Interagieren Sie mit der Anwendung und führen Sie die zu profilierenden Aktionen aus.

5 Klicken Sie auf die Schaltfläche „Leistungsprofil erfassen“.

Der Zeitraum zwischen dem Klicken auf „Leistungsdaten zurücksetzen“ und dem Klicken auf „Leistungsprofil erfassen“ ist die Interaktionsperiode. Wenn Sie nicht auf die Schaltfläche „Leistungsdaten zurücksetzen“ klicken, schließt das Leistungsprofil alle erfassten Daten seit dem Starten der Anwendung mit ein.

6 Doppelklicken Sie in der Profilansicht auf das Leistungsprofil.

Das folgende Beispiel zeigt die Leistungsprofilansicht:


Objektstatistiken
Leistungsprofil
http://localhost:8500/DataMgtCF/DataMgtCF.html (10.02.2010 15:29:01 - 10.02.2010 15:29:13)
Methode
Paket ...
Modulaufrufe
Kumulative Zeit (ms)
Eigene Zeit (ms)
Durchschn. ...
Durchschn. ...
<anonymous> 1 (0.14%) 40 (4.89%) 0 (0.0%) 40.0 0.0
AbstractServiceWrapper.set ... com.ad... 1 (0.14%) 1 (0.12%) 0 (0.0%) 1.0 0.0
DataMgtCF.DataMgtCF 1 (0.14%) 14 (1.71%) 0 (0.0%) 14.0 0.0
DataMgtCF._DataMgtCF_Arr... 1 (0.14%) 13 (1.59%) 0 (0.0%) 13.0 0.0
DataMgtCF._DataMgtCF_But... 1 (0.14%) 1 (0.12%) 0 (0.0%) 1.0 0.0
DataMgtCF._DataMgtCF_Call... 1 (0.14%) 1 (0.12%) 0 (0.0%) 1.0 0.0
DataMgtCF._DataMgtCF_Dat... 1 (0.14%) 12 (1.47%) 0 (0.0%) 12.0 0.0
DataMgtCF._DataMgtCF_Dat... 1 (0.14%) 1 (0.12%) 0 (0.0%) 1.0 0.0
DataMgtCF._DataMgtCF_Dat... 1 (0.14%) 1 (0.12%) 0 (0.0%) 1.0 0.0
DataMgtCF._DataMgtCF_Em... 1 (0.14%) 9 (1.1%) 0 (0.0%) 9.0 0.0
DataMgtCF.__dg_creationCo... 1 (0.14%) 45 (5.5%) 0 (0.0%) 45.0 0.0

Die folgende Tabelle beschreibt die Spalten in der Leistungsprofilansicht:

Spalte	Beschreibung
Methode	Der Name der Methode und die Klasse, zu der die Methode gehört. Interne, von Flash Player ausgeführte Aktionen werden in eckigen Klammern angezeigt, z. B. `[mark]` und `[sweep]`. Das Verhalten dieser internen Aktionen können Sie nicht ändern. Sie können aber die zugehörigen Informationen für die Profiling- und Optimierungsarbeit nutzen. Weitere Informationen zu diesen Aktionen finden Sie unter „Funktionsweise des Profilers“ auf Seite 164.
Paket	Das Paket, in dem sich die Klasse befindet. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Aufrufe	Die Häufigkeit, mit der die Methode während der Interaktionsperiode aufgerufen wurde. Wenn eine Methode im Vergleich zu anderen Methoden unverhältnismäßig oft aufgerufen wird, sollten Sie überlegen, wie Sie sie optimieren können, um die Ausführungszeit zu reduzieren.
Kumulative Zeit	Die Gesamtzeit (in Millisekunden) zur Ausführung aller Aufrufe dieser Methode sowie aller Aufrufe an nachfolgende Methoden während der Interaktionsperiode.

Spalte	Beschreibung
Eigene Zeit	Die Zeit (in Millisekunden), die zur Ausführung aller Aufrufe dieser Methode während der Interaktionsperiode benötigt wurde.
Durchschnittl. Kumulative Zeit	Die durchschnittliche Zeit (in Millisekunden), die zur Ausführung aller Aufrufe dieser Methode sowie der Aufrufe nachfolgender Methoden während der Interaktionsperiode benötigt wurde.
Durchschnittliche eigene Zeit	Die durchschnittliche Zeit (in Millisekunden), die diese Methode für die Ausführung während der Profiling-Periode brauchte.

Wenn Sie in der Leistungsprofilansicht auf eine Methode doppelklicken, zeigt Flex in der Methodenstatistikenansicht Informationen zu dieser Methode an. Dort können Sie weitere Details zum Call-Stack der betreffenden Methode anzeigen. Weitere Informationen finden Sie unter „Leistungsmerkmale von Methoden erkennen“ auf Seite 159.

Zur Einschränkung der Daten in der Leistungsprofilansicht verwenden Sie die Profilerfilter. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.

Leistungsprofile können zur späteren Verwendung abgespeichert werden. Weitere Informationen finden Sie unter „Profiling-Daten speichern und laden“ auf Seite 170.

Leistungsmerkmale von Methoden erkennen

Die Methodenstatistikenansicht zeigt die Leistungsmerkmale der ausgewählten Methode an. In der Regel nutzen Sie die Methodenstatistikenansicht zum Identifizieren von Leistungsengpässen in der Anwendung. Indem Sie sich beispielsweise die Ausführungszeiten einer Methode anzeigen lassen, können Sie sehen, welche Methoden unverhältnismäßig viel Zeit beanspruchen. Diese Methoden können Sie dann gezielt optimieren.

Weitere Informationen finden Sie unter „Leistungsprofilansicht“ auf Seite 157.

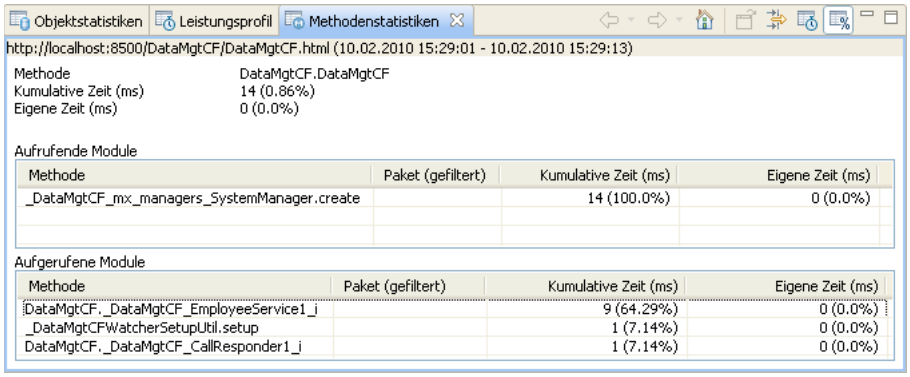
Methodendetails in der Methodenstatistikenansicht anzeigen

1. Doppelklicken Sie in der Leistungsprofilansicht auf eine Zeile oder wählen Sie in der Leistungsprofilansicht eine Methode aus.
2. Klicken Sie auf die Schaltfläche „Methodenstatistiken öffnen“.

Die Ansicht enthält drei Bereiche:

- Eine Zusammenfassung der Leistung der ausgewählten Methode, einschließlich der Anzahl der Aufrufe, der kumulativen Zeit und der eigenen Zeit.
- Aufrufende Methoden-Tabelle: Details über die Methoden, die die ausgewählte Methode aufgerufen haben. In einigen Situationen ist es wichtig zu wissen, ob die ausgewählte Methode zu häufig aufgerufen wird, wie sie verwendet und ob sie richtig verwendet wird.
- Aufgerufene Module: Details über die Methoden, die von der ausgewählten Methode aufgerufen wurden.

Das folgende Beispiel zeigt die Zusammenfassung der Methoden sowie die aufrufende und aufgerufene Methoden-Tabellen der Methodenstatistikenansicht:


Objektstatistiken
Leistungsprofil
Methodenstatistiken
http://localhost:8500/DataMgtCF/DataMgtCF.html (10.02.2010 15:29:01 - 10.02.2010 15:29:13)
Methode DataMgtCF.DataMgtCF
Kumulative Zeit (ms) 14 (0.86%)
Eigene Zeit (ms) 0 (0.0%)
Aufrufende Module
Methode | Paket (gefiltert) | Kumulative Zeit (ms) | Eigene Zeit (ms)
_DataMgtCF_mx_managers_SystemManager.create | 14 (100.0%) | 0 (0.0%)
Aufgerufene Module
Methode | Paket (gefiltert) | Kumulative Zeit (ms) | Eigene Zeit (ms)
DataMgtCF._DataMgtCF_EmployeeService1_i | 9 (64.29%) | 0 (0.0%)
_DataMgtCFWatcherSetupUtil.setup | 1 (7.14%) | 0 (0.0%)
DataMgtCF._DataMgtCF_CallResponder1_i | 1 (7.14%) | 0 (0.0%)

Die folgende Tabelle beschreibt die Felder in der aufrufenden Methoden-Tabelle der Methodenstatistikenansicht:

Spalte	**Beschreibung**
Methode	Die Methoden, die die Methode aufgerufen haben, die in der Zusammenfassung oben in dieser Ansicht erscheint. Ist diese Liste leer, wurde die Zielmethode nicht durch eine der nicht herausgefilterten Methoden aufgerufen.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Kumulative Zeit	Die Zeit (in Millisekunden), die für die Ausführung jeder aufrufenden Methode und aller nachfolgenden Methoden gebraucht wurde.
Eigene Zeit	Die Zeit (in Millisekunden), die jede aufrufende Methode für die Ausführung gebraucht hat. Dies schließt keine Methoden ein, die von nachfolgenden Methoden aufgerufen wurden.

Die folgende Tabelle beschreibt die Felder in der aufgerufenen Methoden-Tabelle der Methodenstatistikenansicht.

Spalte	**Beschreibung**
Methode	Die Methoden, die von der Methode aufgerufen wurden, die in der Zusammenfassung oben in dieser Ansicht angezeigt wird. Ist diese Liste leer, wurde die Zielmethode nicht durch eine der nicht herausgefilterten Methoden aufgerufen.
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Name der Datei, in der sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Wenn das Feld „Paket“ leer ist, gehört die Klasse zum globalen oder zum nicht benannten Paket.
Kumulative Zeit	Die Zeit (in Millisekunden), die jede aufgerufene Methode und alle nachfolgenden Methoden für die Ausführung gebraucht haben.
Eigene Zeit	Die Zeit (in Millisekunden), die jede aufgerufene Zeit für die Ausführung gebraucht hat. Dies schließt keine Methoden ein, die von nachfolgenden Methoden aufgerufen wurden.

Sie können im Call-Stack navigieren, während Sie versuchen, die Leistungsengpässe zu finden, indem Sie entweder in der aufrufenden oder der aufgerufenen Methoden-Tabelle auf die Methoden klicken. Wenn Sie in diesen Tabellen auf eine Methode doppelklicken, zeigt der Profiler die betreffende Zusammenfassung oben in der Methodenstatistikenansicht an und präsentiert dann die aufrufenden und die aufgerufenen Methoden für die neu ausgewählte Methode in den beiden Tabellen.

Hinweis: *Die kumulative Zeit minus die eigene Zeit in dieser Ansicht entspricht nicht immer der kumulativen Zeit der aufrufenden Methoden. Der Grund hierfür ist, dass die kumulative Zeit beim Drillup-Verfahren zu einer aufrufenden Methode die eigene Zeit plus alle Ketten, von denen die Originalmethode aufgerufen wurde, ist, jedoch nicht die anderer aufgerufener Methoden.*

Zum Navigieren im Call-Stack können Sie auch die Profiler-Schaltflächen „Zurück“, „Weiter“ und „Zum Anfang“ verwenden.

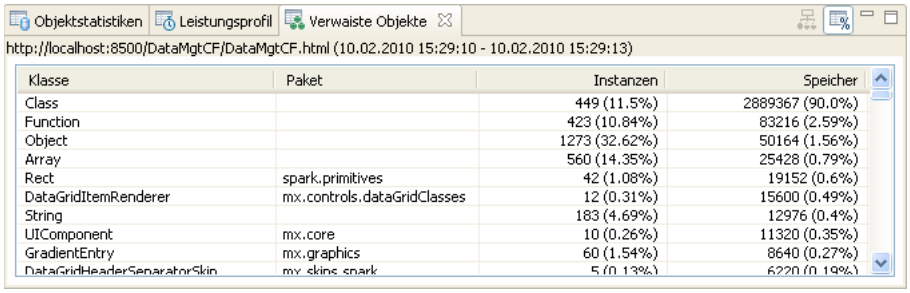
Zur Einschränkung der Daten in der Methodenstatistikenansicht können Sie die Profilerfilter verwenden. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.

Verwaiste-Objekte-Ansicht

Die Verwaiste-Objekte-Ansicht zeigt die Unterschiede zwischen zwei Speicherabbildern der Anwendung, die profiliert wird. Die Unterschiede in dieser Ansicht sind die Anzahl der Objektinstanzen im Speicher und die von diesen Objekten belegte Speichermenge. Dies ist hilfreich zum Ermitteln von Speicherlecks. Die Zeit zwischen zwei Speicherabbildern ist das so genannte *Speicherabbildintervall*.

Zum Öffnen der Verwaiste-Objekte-Ansicht wählen Sie zwei Speicherabbilder aus und klicken auf die Schaltfläche „Verwaiste Objekte“. Weitere Informationen zur Aufzeichnung eines Speicherabbilds finden Sie unter „Speicherabbildansicht“ auf Seite 151.

Das folgende Beispiel zeigt die Verwaiste-Objekte-Ansicht:



Die folgende Tabelle beschreibt die Spalten in der Verwaiste-Objekte-Ansicht:

Spalte	Beschreibung
Klasse	Die Klassen, die während des Speicherabbildintervalls erstellt, aber nicht gelöscht wurden.

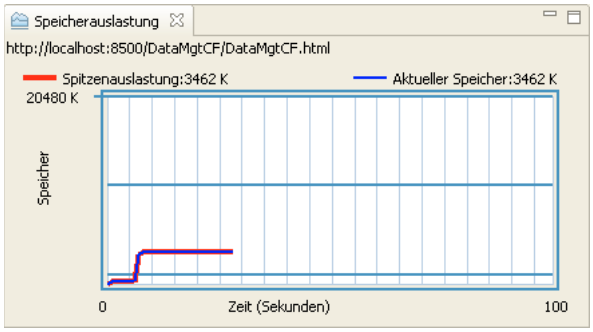
Spalte	Beschreibung
Paket	Das Paket, in dem sich die einzelnen Klassen jeweils befinden. Wenn die Klasse sich nicht in einem Paket befindet, ist der Wert dieses Felds der Dateiname, in dem sich die Klasse befindet. Die Zahl nach dem Dollarzeichen ist eine eindeutige ID der betreffenden Klasse. Ist das Feld „Paket“ leer, befindet sich das Objekt im globalen oder im unbenannten Paket.
Instanzen	Die Anzahl Instanzen, die während des Speicherabbildintervalls erstellt wurden. Dies ist der Unterschied zwischen der Anzahl Instanzen jeder Klasse im ersten Speicherabbild und im zweiten. Angenommen, es gab im ersten Abbild 22.567 Strings und im zweiten 22.861, dann ist der Wert dieses Felds 294.
Speicher	Die Menge Speicherplatz, die während des Speicherabbildintervalls zugewiesen wurde. Dies ist der Unterschied zwischen der Menge Speicher, die die Instanzen jeder Klasse zur Zeit des ersten Speicherabbilds und zur Zeit des zweiten belegten. Angenommen, die Strings belegten 2.031.053 Byte im ersten Abbild und 2.029.899 Byte im zweiten, dann ist der Wert dieses Felds 1154 Byte.

Weitere Informationen zur Ermittlung von Speicherlecks finden Sie unter „Speicherlecks finden“ auf Seite 173.

Speicherauslastungsdiagramm

Das Speicherauslastungsdiagramm zeigt den Speicher, der von der profilierten Anwendung belegt wird. Dieser Wert ist nicht mit der Speicherauslastung des Flash Player und dessen Browser identisch. Der Grund hierfür ist, dass dieser Wert weder den Profiler-Agenten noch die Speicherbelegung des Browsers berücksichtigt. Er besteht nur aus der Summe der Live-Objekte der profilierten Anwendung. Wenn Sie also den Speicherauslastungswert in diesem Diagramm mit dem vom Browser belegten Speicher vergleichen (z. B. im Windows Task Manager), erhalten Sie unterschiedliche Ergebnisse.

Die folgende Abbildung zeigt das Diagramm in der Speicherauslastungsansicht:



Der Wert für den aktuellen Speicher ist identisch mit der Summe der Gesamtwerte in der Speicherspalte der Live-Objekte-Ansicht, sofern alle Filter deaktiviert sind.

Der Spitzenwert ist die höchste Speicherauslastung, die von der Anwendung während der aktuellen Profilingsitzung beansprucht wurde.

Das Speicherauslastungsdiagramm zeigt die Speicherbelegung für die letzten 100 Sekunden. Diese Zahl lässt sich nicht konfigurieren, und Sie können für das Diagramm keine historischen Daten speichern.

Wenn Sie das Speicherauslastungsdiagramm schließen und es dann wieder öffnen möchten, klicken Sie auf die Dropdownliste in der Profilansicht und wählen Sie „Speicherauslastung“ aus.

Profiling einer Anwendung

Der Profiler hilft Ihnen beim Erkennen von Leistungsengpässen und Speicherlecks in Anwendungen. Sie starten den Profiler über Adobe Flash Builder und während Sie mit der Anwendung interagieren, zeichnet der Profiler Daten zum Status der Anwendung auf. Die aufgezeichneten Daten umfassen die Anzahl der Objekte, die Größe dieser Objekte, die Anzahl der Methodenaufrufe und die für diese verwendete Zeit.

Durch das Profiling einer Anwendung gelangen Sie zu einem besseren Verständnis Ihrer Anwendung. Dazu gehören die folgenden Aspekte:

Aufrufhäufigkeit In einigen Fällen werden rechenintensive Methoden mehrmals aufgerufen, obwohl dies nicht notwendig ist. Durch die Identifizierung der am häufigsten aufgerufenen Methoden können Sie Leistungsoptimierungen auf spezifische Bereiche der Anwendung konzentrieren, um die größtmögliche Optimierung zu erzielen.

Methodendauer Über den Profiler können Sie erfahren, wie viel Zeit für eine bestimmte Methode aufgewendet wurde bzw., wenn die Methode mehrfach aufgerufen wird, welches die durchschnittliche Dauer innerhalb des profilierten Zeitraums war. Wenn sich herausstellt, dass einige Methoden einen Leistungsengpass verursachen, können Sie versuchen, diese Methoden zu optimieren.

Call-Stacks (Aufrufstapel) Durch das Verfolgen des Call-Stacks einer Methode können Sie den vollständigen Pfad der Anwendung beim sequenziellen Aufrufen der Methoden nachvollziehen. Manchmal stellen Sie hierbei fest, dass einige Methoden unnötig oft aufgerufen werden.

Anzahl von Instanzen (Objektzuweisung) Unter Umständen stellen Sie fest, dass ein und dasselbe Objekt mehrere Male erstellt wird, obwohl nur eine ganz bestimmte Anzahl von Instanzen erforderlich ist. In diesen Fällen könnten Sie sich überlegen, ein Singleton-Muster zu implementieren, wenn Sie wirklich nur eines dieser Objekte benötigen. Sie können auch andere Maßnahmen ergreifen, um überflüssige Objektzuweisungen zu verhindern. Wenn die Anzahl der Objekte groß, aber unumgänglich ist, können Sie unter Umständen das Objekt selbst optimieren, um so die Gesamtressourcen und die Speicherplatzbelegung zu verringern.

Objektgröße Wenn Sie feststellen, dass einige Objekte unverhältnismäßig groß sind, können Sie versuchen, sie zu optimieren, damit sie weniger Speicherplatz belegen. Besonders hilfreich ist dies bei der Optimierung von Objekten, die viele Male in der Anwendung erstellt werden.

Garbage Collection Beim Vergleichen von Profilingschnappschüssen stellen Sie möglicherweise fest, dass einige inzwischen „verwaiste“ Objekte noch immer in der Anwendung existieren oder im Speicher verbleiben. Um diese Speicherlecks zu verhindern, fügen Sie eine Logik hinzu, die alle noch verbleibenden Verweise auf diese Objekte entfernt.

Das Profiling ist ein iterativer Prozess und ist Bestandteil jeder Anwendungsentwicklungsphase. Um schnell problematische Bereiche zu erkennen und den größtmöglichen Nutzen aus dem Profiling zu ziehen, sollten Sie die Anwendung während der Entwicklung frühzeitig und oft profilieren.

Profiling-Arten

Ehe Sie den Profiler einsetzen, sollten Sie sich entscheiden, ob Sie ein Leistungs- oder Speicher-Profiling durchführen möchten.

Beim *Leistungs-Profiling* wird in der Anwendung nach Methoden gesucht, deren Ausführung langsam läuft und die verbesserungswürdig sind. Wenn diese Methoden identifiziert sind, können ihre Ausführungszeiten optimiert werden. Dies bewirkt, dass die Anwendung schneller läuft und auch schneller auf Benutzerinteraktion reagiert. Das Leistungs-Profiling ermittelt im Allgemeinen zwei besondere Methoden: solche, die nur einmal aufgerufen werden, aber mehr Zeit beanspruchen als vergleichbare andere Methoden, und solche, die an sich nicht viel Zeit beanspruchen, jedoch häufig aufgerufen werden. Die Leistungs-Profilingdaten dienen dazu, diese Methoden zu ermitteln und dann zu optimieren. Es ist durchaus möglich, dass sich eine Verringerung der Anzahl der Aufrufe einer Methode als effektivere Lösung als ein Refaktorieren des Programmcodes innerhalb der Methode erweist.

Beim *Speicher-Profiling* wird untersucht, wie viel Speicher jedes Objekt bzw. jeder Objekttyp in der Anwendung belegt. Die Speicher-Profilingdaten werden auf verschiedene Arten genutzt: Um Objekte zu finden, die größer sind als notwendig, um zu ermitteln, ob es zu viele Objekte eines bestimmten Typs gibt, und um festzustellen, ob es Objekte gibt, die nicht durch den Garbagekollektor bereinigt werden (Speicherlecks). Anhand der Speicher-Profilingdaten können Sie versuchen, die Größe der Objekte zu reduzieren, die Anzahl der erstellten Objekte zu verringern oder Objekte durch Entfernen von Verweisen auszuräumen.

Das Speicher-Profiling kann die Leistung einer Anwendung beeinträchtigen, da sie wesentlich mehr Speicher als das Leistungs-Profiling benötigt. Führen Sie Speicher-Profiling nur bei Bedarf durch.

Vielfach werden sowohl ein Leistungs- als auch ein Speicher-Profiling durchgeführt, um die Ursache von Leistungsproblemen zu ergründen. Das Leistungs-Profiling dient dazu, Methoden zu identifizieren, die zu übermäßigen Speicherbelegungen und langen Ausführungszeiten Anlass geben. Anschließend wird das Speicher-Profiling eingesetzt, um Speicherlecks in diesen Methoden festzustellen.

Sobald Sie sich für eine der Profiling-Arten entschieden haben, können Sie den Profiler starten.

Zusätzliche Ressourcen

Der Profiler allein verbessert die Größe, Geschwindigkeit und wahrnehmbare Leistung der Anwendung nicht. Nachdem Sie den Profiler für die Ermittlung der problematischen Methoden und Klassen eingesetzt haben, lesen Sie in der Flex-Dokumentation nach, wie Sie die Anwendung optimieren können:

- Optimieren von Anwendungen
- Verbesserung der Leistung beim Start

Funktionsweise des Profilers

Der Profiler ist ein Agent, der mit der in Flash Player ausgeführten Anwendung kommuniziert. Die Verbindung zur Anwendung erfolgt über einen lokalen Socket. Aus diesem Grund müssen Sie möglicherweise Ihre Virenschutzsoftware deaktivieren (wenn diese Socket-Kommunikation unterbindet).

Während der Profiler läuft, erstellt er in kurzen Abständen regelmäßige Schnappschüsse der Daten und protokolliert die Aktivitäten des Adobe Flash Player. Dies wird als *Sampling* bezeichnet. Wenn die Anwendung beispielsweise im Zeitpunkt des Abbilds (Snapshot) eine Methode ausführt, wird diese Methode vom Profiler protokolliert. Wenn die Anwendung beim nächsten Speicherabbild noch immer mit der Ausführung derselben Methode beschäftigt ist, protokolliert der Profiler weiterhin die Zeit. Wenn die Anwendung im Zeitpunkt des nächsten Speicherabbilds zur nächsten Methode gewechselt hat, kann der Profiler die Ausführungsdauer der Methode protokollieren.

Das Sampling gestattet eine Profilierung ohne merkliche Verlangsamung der Anwendung. Das Intervall wird als *Samplingrate* bezeichnet. Typischerweise erfolgt ein Sampling alle 1 ms während der Profilierungsperiode. Dies bedeutet, dass nicht alle Methoden aufgezeichnet werden und dass nicht jedes Speicherabbild bis auf die Millisekunde genau ist. Trotzdem erhalten Sie eine wesentlich bessere Vorstellung davon, welche Methoden unverhältnismäßig länger dauern als andere.

Durch das Parsen der Samplingdaten kann der Profiler jede Methode in der Anwendung zeigen, und der Profiler protokolliert die Ausführungszeiten aller Methoden. Der Profiler protokolliert auch die Speichernutzung und die Stacktraces und zeigt die Daten in einer Reihe von Ansichten oder Bedienfeldern auf. Methodenaufrufe werden nach Ausführungszeiten und Anzahl von Aufrufen organisiert, aber auch nach der Anzahl der in der Methode erstellten Objekte.

Der Profiler berechnet außerdem auch kumulative Datenwerte. Wenn Sie beispielsweise Methodenstatistiken anzeigen, schließen die kumulativen Daten die Zeit und den zugewiesenen Speicher für die betreffende Methode ein, zusammen mit der Zeit und Speicherzuweisung während der Ausführungsdauer aller Methoden, die aus dieser Methode aufgerufen wurden. Sie können mithilfe von Drilldowns in weitere Methodenaufrufe so lange suchen, bis Sie die Ursache der Leistungsprobleme finden.

Profiling-APIs

Der Profiler benutzt die ActionScript-APIs, die im flash.sampler.*-Paket definiert sind. Dieses Paket enthält die Beispiel-, StackFrame-, NewObjectSample- und DeleteObjectSample-Klassen. Sie können die Methoden und Klassen in diesem Paket zum Schreiben einer eigenen Profiler-Anwendung benutzen oder eine Untermenge der Profilierungsfunktionalität in die Anwendungen übernehmen.

Zusätzlich zu den Klassen im flash.sampler.*-Paket verwendet der Profiler auch Methoden der System-Klasse.

Interne Player-Aktionen

Der Profiler zeichnet normalerweise Daten zu Methoden einer bestimmten Klasse auf, die der Flash Player beim Sampling des Abbilds ausgeführt hat. Manchmal zeichnet der Profiler aber auch interne Player-Aktionen auf. Diese Aktionen sind durch eckige Klammern gekennzeichnet. Beispiele hierfür sind `[keyboardEvent]`, `[mark]` und `[sweep]`.

Wenn in der Methodenliste beispielsweise `[keyboardEvent]` mit einem Wert von 100 angegeben ist, wissen Sie, dass der Player während der Interaktionsperiode mindestens 100 Mal eine interne Aktion im Zusammenhang mit diesem Ereignis ausgeführt hat.

Die folgende Tabelle beschreibt die internen Flash Player-Aktionen, die in Profilingdaten erscheinen.

Aktion	Beschreibung
`[generate]`	Der Just-in-Time (JIT)-Compiler generiert AS3-Maschinencode.
`[mark]`	Der Flash Player markiert Live-Objekte für den Garbagekollektor.
`[newclass]`	Der Flash Player definiert eine Klasse. Normalerweise geschieht dies beim Start, eine neue Klasse kann jedoch jederzeit geladen werden.
`[pre-render]`	Der Flash Player bereitet das Rendering von Objekten vor (einschließlich geometrischer Berechnungen sowie das Durchlaufen der Anzeigeliste vor dem Rendering).
`[reap]`	Der Flash Player gewinnt alle DRC (Deferred Reference Counting)-Objekte zurück.
`[render]`	Der Flash Player rendert Objekte in der Anzeigeliste (pixelweise).

Aktion	Beschreibung
`[sweep]`	Der Flash Player gewinnt den Speicherplatz, der von nicht markierten Objekten belegt ist, zurück.
`[verify]`	Der JIT-Compiler führt eine ActionScript 3.0 Bytecode-Überprüfung aus.
`[event_typeEvent]`	Der Flash Player löst das vorgegebene Ereignis aus.

Diese Informationen helfen Ihnen, Leistungsprobleme zu identifizieren. Wenn Sie beispielsweise sehr viele Einträge für `[mark]` und `[sweep]` sehen, können Sie davon ausgehen, dass viele Objekte erstellt und dann für die Garbagekollektor vorgemerkt werden. Indem Sie diese Zahlen in verschiedenen Leistungsprofilen miteinander vergleichen, können Sie feststellen, ob die vorgenommenen Änderungen die gewünschte Wirkung zeigen.

Um die Daten zu diesen internen Aktionen einzusehen, zeigen Sie in der Leistungsprofilansicht ein Leistungsprofil bzw. in der Speicherreservierungsansicht ein Speicherprofil an. Weitere Informationen finden Sie unter „Leistungsprofilansicht“ auf Seite 157 und „Speicherreservierungsansicht“ auf Seite 155.

Der Profiler

Der Profiler erfordert Flash Player Version 9.0.124 oder höher. Sie können Anwendungen profilieren, die für Flex 2, Flex 2.0.1 und Flex 3 kompiliert wurden. Sie können den Profiler zum Profilieren von ActionScript 3.0-Anwendungen einsetzen, die mit Flash Authoring erstellt wurden, sowie für Desktop-Anwendungen, die mit Adobe® AIR® ausgeführt werden.

Der Profiler benötigt Debuginformationen der Anwendung, die profiliert wird. Wenn Sie eine Anwendung kompilieren und den Profiler starten, nimmt Flash Builder die Debuginformationen standardmäßig mit in die Anwendung auf. Um die Debuginformationen explizit in eine Anwendung aufzunehmen, setzen Sie die Compiler-Option `debug` auf `true`. Wenn Sie eine Anwendung über die Option „Releasebuild exportieren“ exportieren, enthält die Anwendung keine Debuginformationen.

Profiler-Symbolleiste

Die folgende Tabelle beschreibt die Schaltflächen in der Profiler-Symbolleiste:

Schaltfläche	Name	Beschreibung
	Fortsetzen	Setzt die Profilingsitzung fort. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt und die Anwendung derzeit unterbrochen ist.
	Unterbrechen	Unterbricht die Profilingsitzung. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt ist und die Anwendung derzeit ausgeführt wird.
	Beenden	Beendet die Profilingsitzung. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt und die Anwendung nicht bereits beendet wurde.
	Garbagekollektor ausführen	Weist Flash Player zur Ausführung einer automatischen Speicherbereinigung (Garbage Collection) an. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt ist und die Anwendung derzeit ausgeführt wird. Weitere Informationen zur Speicherbereinigung (Garbage Collection) finden Sie unter „Garbage Collection“ auf Seite 172.

Schaltfläche	Name	Beschreibung
	Speicherabbild erstellen	Speichert die Speicherauslastung einer Anwendung, damit Sie sie überprüfen oder mit anderen Abbildern vergleichen können. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt ist, die betreffende Anwendung derzeit ausgeführt wird und Sie im Start-Dialogfeld die Option „Speicher-Profiling aktivieren“ aktiviert haben. Der Profiler fügt in der Profilansicht neue Speicherabbilder als untergeordnete Elemente der ausgewählten Anwendung hinzu. Zum Öffnen des neuen Speicherabbilds in der Speicherabbildansicht doppelklicken Sie auf den Speicherabbildeintrag. Die Garbage Collection erfolgt automatisch vor der Aufzeichnung von Speicherabbildern. Mit anderen Worten: Ein Klick auf die Schaltfläche „Speicherabbild erstellen“ ist gleichbedeutend mit einem Klicken auf „Garbagekollektor ausführen“, gefolgt von „Speicherabbild erstellen“. Weitere Informationen zu Speicherabbildern finden Sie unter „Speicherabbildansicht“ auf Seite 151.
	Verwaiste Objekte suchen	Vergleicht zwei Speicherabbilder in der Verwaiste-Objekte-Ansicht. Diese Option ist nur aktiviert, wenn zwei Speicherabbilder ausgewählt sind und Sie im Start-Dialogfeld die Option „Speicher-Profiling aktivieren“ aktiviert haben. Weitere Informationen zur Verwaiste-Objekte-Ansicht finden Sie unter „Verwaiste-Objekte-Ansicht“ auf Seite 161.
	Speicherreservierung einblenden	Vergleicht die Methoden zwischen zwei Speicherabbildern in der Speicherreservierungsansicht. Diese Option ist nur aktiviert, wenn zwei Speicherabbilder ausgewählt sind und Sie im Start-Dialogfeld die Option „Speicher-Profiling aktivieren“ aktiviert haben. Weitere Informationen zur Speicherreservierungsansicht finden Sie unter „Speicherreservierungsansicht“ auf Seite 155.
	Leistungsdaten zurücksetzen	Löscht die Leistungs-Profilingdaten. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt ist und die Anwendung ausgeführt wird und Sie im Start-Dialogfeld die Option „Leistungs-Profiling aktivieren“ aktiviert haben. In der Regel klicken Sie auf diese Schaltfläche, interagieren mit der Anwendung und klicken dann auf „Leistungsprofil erfassen“, um ein Leistungsabbild der Anwendung ab dem Zeitpunkt der Rücksetzung der Daten zu erhalten. Weitere Informationen zur Leistungsprofilansicht finden Sie unter „Leistungsprofilansicht“ auf Seite 157.

Schaltfläche	Name	Beschreibung
	Leistungsprofil erfassen	Speichert ein neues Leistungsabbild als untergeordnetes Objekt der ausgewählten Anwendung. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt ist und die Anwendung ausgeführt wird und Sie im Start-Dialogfeld die Option „Leistungs-Profiling aktivieren“ aktiviert haben. Zum Öffnen der Leistungsprofilansicht doppelklicken Sie auf den Eintrag des Leistungsabbilds. Weitere Informationen zur Leistungsprofilansicht finden Sie unter „Leistungsprofilansicht“ auf Seite 157.
	Löschen	Entfernt die Daten des ausgewählten Abbilds aus dem Speicher. Wenn Sie auf diese Schaltfläche klicken, wird auch die Anwendung aus der Profilansicht entfernt, sofern sie beendet wurde. Diese Option ist nur aktiviert, wenn ein Leistungs- oder ein Speicherabbild ausgewählt wurde.
–	Speichern	Speichert Profilingdaten auf der Festplatte. Der Befehl „Speichern“ ist über das Ansichtsmenü des Profilers verfügbar. Diese Option ist nur aktiviert, wenn ein Anwendungsname ausgewählt wurde.

Einige Ansichten im Profiler wie z. B. Methodenstatistiken und Objektstatistiken haben Navigationsschaltflächen, die zum Durchlaufen des Stapels oder zum Ändern der Ansicht verwendet werden können. Die folgende Tabelle beschreibt die Navigationsschaltflächen in diesen Profileransichten:

Schaltfläche	Name	Beschreibung
	Zurück	Zeigt alle Methoden, die Sie von der ersten ausgewählten Methode bis zur derzeit angezeigten durchlaufen haben.
	Weiter	Zeigt die aktuell angezeigte Methode und die Methoden, die zu der aktuell ausgewählten führen. Dieses Element ist nur aktiviert, nachdem Sie zu einer Methode zurückgegangen sind.
	Home	Zeigt die erste ausgewählte Methode an.
	Quelldatei öffnen	Öffnet einen Quelleditor, der den Quellcode der ausgewählten Methode anzeigt.
	Filter	Hiermit können Sie steuern, welche Methoden Sie in der Tabelle einschließen wollen. Weitere Informationen finden Sie unter „Profiler-Filter“ auf Seite 176.
	Nullzeitmethoden ein-/ausblenden	Blendet Methoden ein oder aus, die in der Spalte mit den Durchschnittszeiten einen Wert von 0.00 aufweisen (dies bedeutet, dass sie nicht in den Samples erscheinen).

Profiler starten, stoppen und fortsetzen

Sie können Anwendungen profilieren, die Sie gerade in Flash Builder entwickeln. Flash Builder nimmt beim Kompilieren und Ausführen einer Anwendung während einer Profilingsitzung Debuginformationen mit auf. Sie können auch externe Anwendungen profilieren, die Sie zwar nicht momentan in Flash Builder entwickeln, aber deren SWF-Datei über eine URL oder ein Dateisystem verfügbar ist. Um eine Anwendung zu profilieren, muss die SWF-Anwendungsdatei Debuginformationen enthalten. Weitere Informationen finden Sie unter „Externe Anwendungen profilieren“ auf Seite 171.

Profiling einer Anwendung starten

1 Schließen Sie alle Browserinstanzen.

2 Öffnen Sie die Anwendung in Flash Builder.

3 Klicken Sie in der Hauptsymbolleiste auf die Schaltfläche „*Anwendungsname* profilieren“. Flash Builder fordert Sie zum Schließen aller noch geöffneten Browserinstanzen auf, sofern Sie dies noch nicht getan haben.

4 Klicken Sie auf „OK“. Flash Builder kompiliert die Anwendung und startet sie in einem separaten Browserfenster. Flash Builder zeigt außerdem das Dialogfeld „Profiler konfigurieren“ an.

5 Wählen Sie die gewünschten Optionen im Dialogfeld „Profiler konfigurieren“ aus und klicken Sie auf „Fortsetzen“. Um eine Anwendung zu profilieren, müssen Sie die Option „Speicher-Profiling aktivieren“ bzw. „Leistungs-Profiling aktivieren“ aktivieren. Wenn Sie eine Anwendung profilieren, können Sie beide Optionen auswählen.

Die folgende Tabelle beschreibt die Optionen:

Option	Beschreibung
Verbunden von	Der Server, von dem die Anwendung gestartet wird. Wenn die Anwendung auf dem gleichen Computer wie der Profiler ausgeführt wird, ist dieser Wert „localhost“. Dieser Wert kann nicht geändert werden. Sie können aber eine Anwendung profilieren, die auf einem anderen Computer ausgeführt wird. Siehe „Externe Anwendungen profilieren“ auf Seite 171.
Anwendung	Die Anwendung, die profiliert werden soll. Dieser Wert kann nicht geändert werden.
Speicher-Profiling aktivieren	Weist den Profiler zum Erfassen von Speicherdaten an. Diese Option dient zur Ermittlung von Speicherlecks und unnötig erstellten Objekten. Beim Durchführen eines Leistungs-Profilings können Sie diese Option deaktivieren. Die Option „Speicher-Profiling aktivieren“ ist standardmäßig aktiviert.
Live-Speicherdaten ansehen	Weist den Profiler an, während des Profilings Speicherdaten in der Live-Objekte-Ansicht anzuzeigen. Dies ist weder beim Speicher- noch beim Leistungs-Profiling erforderlich. Die Option kann nur aktiviert werden, wenn zuvor die Option „Speicher-Profiling aktivieren“ aktiviert wurde. Die Option „Live-Speicherdaten ansehen“ ist standardmäßig aktiviert.
Objektspeicherzuweisung-Stacktraces generieren	Weist den Profiler an, bei jeder Erstellung eines neuen Objekts ein Stacktrace zu erfassen. Wenn diese Option aktiviert wird, verläuft das Profiling langsamer. Sie wählen diese Option normalerweise nur dann aus, wenn Sie sie wirklich benötigen. Diese Option ist nur verfügbar, wenn Sie die Option „Speicher-Profiling aktivieren“ aktivieren. Die Option „Objektspeicherzuweisung-Stacktraces generieren“ ist standardmäßig deaktiviert. Wenn Sie diese Option nicht aktivieren, können Sie die Speicherreservierungsinformationen weder in der Objektverweise- noch in der Speicherreservierungsansicht sehen.
Leistungs-Profiling aktivieren	Weist den Profiler an, in den Sampling-Intervallen Stracktrace-Daten zu erfassen. Ermitteln Sie anhand dieser Sample, wo Ihre Anwendung den Großteil der Ausführungszeit verbringt. Beim Speicher-Profiling können Sie diese Option deaktivieren. Die Option „Leistungs-Profiling aktivieren“ ist standardmäßig aktiviert.

Die Standardeinstellungen dieser Optionen können Sie über die Profiling-Voreinstellungen ändern. Weitere Informationen finden Sie unter „Profiler-Voreinstellungen“ auf Seite 171.

6 Sie können jetzt mit der Anwendung interagieren und die Profiler-Daten überprüfen.

Profiling einer Anwendung unterbrechen und fortsetzen

Wenn Sie den Profiler gestartet haben, können Sie Anwendungen in der Profilansicht unterbrechen und fortsetzen. Hierzu wählen Sie zunächst eine Anwendung und dann die gewünschte Aktion aus. Das folgende Beispiel zeigt die Profilansicht mit mehreren Anwendungen. Eine Anwendung läuft gerade und andere wurden beendet.

Profiling einer Anwendung anhalten

1. Wählen Sie die Anwendung in der Profilansicht aus.
2. Klicken Sie auf die Schaltfläche „Beenden“, um die Profilingsitzung manuell zu beenden. Hierdurch wird der Browser weder geschlossen noch der Playerprozess beendet.
3. Um zur Flex-Entwicklungsperspektive zurückzukehren, wählen Sie aus der Dropdownliste „Perspektive“ den Eintrag „Flex-Entwicklung“ aus. Zum Ändern der Perspektive können Sie unter Windows auch Strg+F8 verwenden.

Profiling-Daten speichern und laden

Nach der Ausführung des Profilers können Sie die Daten speichern, damit Sie ein Abbild aus der aktuellen Profilingsitzung mit einem Abbild vergleichen können, das Sie nach Änderungen am Code erstellen. Hierdurch können Sie feststellen, ob Sie die richtigen Problembereiche gefunden haben und ob die Änderungen die Leistung und die Speichernutzung der Anwendung verbessern.

Beim Speichern von Profilingdaten speichern Sie alle Anwendungsdaten in dem betreffenden Profil. Dazu zählen sämtliche Leistungsprofile, Speicherabbilder und Speicherreservierungen. Flash Builder schreibt diese Informationen in eine Gruppe von Binärdateien am von Ihnen festgelegten Speicherort.

Profilingdaten speichern

1. Wählen Sie die Anwendung in der Profilansicht aus.
2. Öffnen Sie die Dropdownliste in der Profilansicht und wählen Sie „Speichern“ aus. Das Dialogfeld zum Suchen eines Ordners wird angezeigt.
3. Wählen Sie einen Ort zum Speichern der Profildaten und klicken Sie auf „OK“. Legen Sie für jeden Satz von Profilingdaten, den Sie speichern möchten, einen neuen Ordner an. Werden die neuen Daten im selben Ordner abgelegt, überschreiben sie die vorhandenen Daten.

Gespeicherte Profilingdaten abrufen

1. Wählen Sie die Ansicht „Gespeicherte Profiling-Daten“ aus.
2. Klicken Sie auf die Schaltfläche „Öffnen“. Das Dialogfeld zum Suchen eines Ordners wird angezeigt.
3. Navigieren Sie zum Ordner, der die Profilingdaten der Anwendung enthält, und klicken Sie auf „OK“. Flash Builder zeigt die verfügbaren Profilingdaten in der Ansicht „Gespeicherte Profiling-Daten“ an. Ein Fortsetzen der Anwendung in dieser Ansicht ist nicht möglich, aber Sie können die Speicherabbilder, die Leistungsprofile und andere gespeicherte Daten anzeigen.

Gespeicherte Anwendungsdaten können nicht von Flash Builder aus gelöscht werden.

Profilingdaten löschen

1. Wählen Sie in der Profilansicht das Speicherabbild der Anwendung aus.
2. Klicken Sie auf die Schaltfläche „Löschen“.

Profiler-Voreinstellungen

Sie können Profilervoreinstellungen festlegen, die dann für alle Profilingsitzungen gelten. Sie können diese Einstellungen verwenden, um den Flash Player/Browser zu definieren, den Sie für das Profiling der Anwendung verwenden, und auch die Standardfilter und die Portnummer der Anwendung festlegen, wenn die profilierte Anwendung auf einem Server läuft.

Flash Builder-Voreinstellungen für mehrere Profilingsitzungen festlegen

❖ Öffnen Sie das Dialogfenster „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Profiler“.

Wählen Sie die gewünschte Kategorie unter dem Eintrag „Profiler“ aus, um die dazugehörigen Optionen anzuzeigen. Die folgende Tabelle beschreibt die möglichen Voreinstellungen:

Menüauswahl	Beschreibung
Profiler	Dient zur Auswahl der Standard-Profiling-Methode. Aktivieren bzw. deaktivieren Sie hier das Speicher- und Leistungs-Profiling.
Verbindungen	Hier können Sie die Nummer des Ports definieren, auf dem Flash Builder die profilierte Anwendung erwartet. Die Standardportnummer ist 9999. Die Portnummer 7935 darf nicht verwendet werden, da sie vom Debugger genutzt wird.
Ausschlussfilter	Hier können Sie die Standardpakete definieren, die von den Profileransichten ausgeschlossen sind. Weitere Informationen zur Verwendung von Filtern finden Sie unter „Profiler-Filter“ auf Seite 176.
Einschlussfilter	Hier können Sie die Standardpakete definieren, die in den Profileransichten eingeschlossen sind. Alle anderen Pakete werden ausgeschlossen. Weitere Informationen zur Verwendung von Filtern finden Sie unter „Profiler-Filter“ auf Seite 176.
Player/Browser	Hier können Sie den Speicherort des Flash Player- und des Browser-Programms definieren, das Flash Builder zum Ausführen der profilierten Anwendung benutzt.

Externe Anwendungen profilieren

Sie können nicht nur interne, in Flash Builder entwickelte Anwendungen, sondern auch externe Anwendungen profilieren. Externe Anwendungen können beliebige SWF-Dateien sein, die sich an einem zugänglichen Ort befinden müssen. Hierzu gehören Anwendungen auf einem Remotewebserver oder im lokalen Dateisystem.

Für die SWF-Datei können Sie entweder eine URL oder einen Speicherort im Dateisystem angeben. Bei Angabe einer URL startet Flash Builder die SWF-Datei der Anwendung im Standardbrowser. Der Browser muss die Debugger-Version von Flash Player verwenden, um die Anwendung erfolgreich zu profilieren.

Wenn Sie für die SWF-Datei einen Ort im Speichersystem angeben, öffnet Flash Builder die Anwendung in der Debugger-Version des eigenständigen Flash Player. In der Regel verwenden Sie eine URL, um die Datei anzufordern. Die Ausführung von Anwendungen in der eigenständigen Flash Player-Version kann unerwartete Ergebnisse bewirken, besonders wenn die Anwendung Remotedienste oder Netzwerkaufrufe verwendet.

Profiling einer externen Anwendung

1 Zur Flash Profiling-Perspektive wechseln

2 Wählen Sie „Profil“ > „Externe Anwendung profilieren“. Das Dialogfeld „Externe Anwendung profilieren“ wird angezeigt.

3 Aktivieren Sie die Option „Ausgewählte Anwendung starten“ (Standardeinstellung) und klicken Sie auf die Schaltfläche „Neu“. Das Dialogfeld „Anwendung hinzufügen“ wird angezeigt.

   Sie können die Anwendung auch manuell starten, indem Sie die Option „Anwendung manuell außerhalb von Flash Builder starten“ aktivieren.

4 Geben Sie den Speicherort der SWF-Datei ein und klicken Sie auf „OK“, oder klicken Sie auf die Schaltfläche „Durchsuchen“ und wählen Sie die Anwendung auf dem Dateisystem aus.

5 Klicken Sie auf die Schaltfläche „Starten“. Wenn Sie für den Speicherort der Anwendung eine URL angegeben haben, startet Flash Builder die Anwendung im Standardbrowser. Wenn Sie einen Speicherort im Dateisystem angegeben haben, öffnet Flash Builder die Anwendung in der eigenständigen Version von Flash Player.

   Wenn Sie eine SWF-Datei angegeben haben, die nicht mit Debuginformationen kompiliert wurde, gibt Flash Builder einen Fehler zurück. Kompilieren Sie die Anwendung erneut mit der Compiler-Option `debug = true` und starten Sie sie erneut.

Garbage Collection

Garbage Collection ist der Vorgang, bei dem der Garbagekollektor Objekte aus dem Speicher entfernt, die nicht länger gebraucht werden. Hierbei wird die Zuweisung von Speicherplatz aufgehoben, der von Instanzen beansprucht wird, auf die nicht mehr verwiesen wird.

Der Flash Player führt eine Garbage Collection durch, wann immer dies im Lebenszyklus einer Anwendung notwendig erscheint. Das Löschen eines Verweises auf ein Objekt löst nicht automatisch eine Garbage Collection aus. Wenn Sie also sämtliche Verweise auf ein Objekt entfernen, hebt der Garbagekollektor nicht unbedingt die ganze Speicherzuweisung für dieses Objekt auf. Das Objekt wird jedoch zu einem Garbage Collection-Kandidaten.

Ein Klick auf die Schaltfläche „Garbagekollektor ausführen“ garantiert nicht, dass sämtliche Objekte, die für eine Garbage Collection infrage kommen, auch wirklich aus dem Speicher entfernt werden. Eine Garbage Collection wird in der Regel durch die Zuweisung von Speicherplatz für neue Ressourcen ausgelöst. Wenn neue Ressourcen Speicher benötigen, der in der aktuellen Zuweisung nicht verfügbar ist, wird der Garbagekollektor ausgeführt. Er setzt den Speicher frei, der dafür vorgemerkt wurde. Dies bedeutet, dass Objekte nicht unbedingt sofort aus dem Speicher bereinigt werden, selbst wenn Sie alle Verweise auf sie entfernen. Die Objekte werden aber sehr wahrscheinlich vom Garbagekollektor entfernt, wenn neue Instanzen erstellt und verwendet werden. Wenn eine Anwendung untätig ist, können Sie ihre Speicherauslastung beobachten. Selbst wenn für den Garbagekollektor vorgemerkte Objekte vorhanden sind, ändert sich die Speicherauslastung einer Anwendung in der Regel erst, wenn eine Interaktion erfolgt.

Der Flash Player weist Speicher in Blöcken von einer bestimmten Menge Byte (nicht byteweise) zu. Wenn ein Teil eines Blocks für den Garbagekollektor vorgemerkt wurde, andere Teile des Blocks jedoch nicht, erfolgt keine Freigabe. Der Garbagekollektor versucht, unbenutzte Teile von Speicherblöcken zu größeren Blöcken zu kombinieren, um diese freizugeben, doch kann dies nicht für jeden Durchlauf gewährleistet werden.

Die Garbage Collection erfolgt automatisch vor der Aufzeichnung von Speicherabbildern. Ein Klick auf die Schaltfläche „Speicherabbild erstellen“ ist also gleichbedeutend mit einem Klick auf die Schaltfläche „Garbagekollektor ausführen“, gefolgt von „Speicherabbild erstellen“.

Garbagekollektor beim Profiling der Anwendung ausführen

❖ Wählen Sie in der Profilansicht die Anwendung aus und klicken Sie auf die Schaltfläche „Garbagekollektor ausführen“.

   Sie können die Wirksamkeit des Garbagekollektors analysieren, indem Sie zwei Speicherabbilder vor und nach der Garbage Collection vergleichen.

Problembereiche ermitteln

Ihnen stehen verschiedene Methoden zur Identifikation von Problembereichen in den Anwendungen mithilfe des Profilers zur Verfügung.

Speicherlecks finden

Eines der häufigsten Probleme bei der Anwendungsentwicklung sind Speicherlecks. Häufig werden Speicherlecks von Objekten verursacht, die während eines bestimmten Zeitraums erstellt, jedoch nicht durch den Garbagekollektor aus dem Speicher entfernt wurden.

Eine Möglichkeit zum Identifizieren von Speicherlecks besteht in der Ermittlung der Anzahl der Verweise. Zu diesem Zweck gehen Sie zur Instanzentabelle in der Objektverweiseansicht. Im Allgemeinen können Sie Verweise von Dokumenteigenschaften und Bindungen ignorieren und sich stattdessen auf unerwartete oder ungewöhnliche Verweise konzentrieren, besonders auf Objekte, die nicht untergeordnete Objekte des betreffenden Objekts sind. Weitere Informationen finden Sie unter „Objektverweiseansicht“ auf Seite 153.

Sie können auch Pfade zu Instanzen eines Objekts überprüfen, um zu ermitteln, ob ein Pfad einen Rückverweis zum Garbagekollektor (GC-Stamm) hat. Eine Instanz, deren Freigabe erwartet wurde, die aber Verweise zum GC-Stamm aufweist, zeigt ein Speicherleck an. In diesen Fällen modifizieren Sie Ihren Anwendungscode, um Verweise auf den GC-Stamm zu entfernen. Eine Instanz, die keine Verweise auf den GC-Stamm aufweist, ist bereit für die Garbage-Collection. Flash Player gibt später diesen Speicher frei.

Eine andere Möglichkeit zum Identifizieren von Speicherlecks besteht im Vergleichen von zwei Speicherabbildern in der Verwaiste-Objekte-Ansicht. Anhand dieses Vergleichs lässt sich feststellen, welche Objekte sich nach einer Abfolge von Ereignissen noch immer im Speicher befinden.

Übliche Maßnahmen zum Aufräumen von Speicherlinks sind die Methoden `disconnect()`, `clearInterval()` und `removeEventListener()`.

Rückverweise zum GC-Stamm für Instanzen suchen

1. Erstellen Sie ein Speicherabbild.

   Siehe „Speicherabbild erstellen und anzeigen“ auf Seite 152.

2. Geben Sie an, wie viele Rückverweispfade gesucht werden sollen.

   Geben Sie in der Speicherabbildansicht an, wie viele Pfade maximal gesucht werden sollen, oder aktivieren Sie die Option „Alle Rückverweispfade anzeigen“.

3. Doppelklicken Sie auf eine Klasse im Speicherabbild, um die Objektverweiseansicht zu öffnen.

4. Erweitern Sie die aufgelisteten Pfade und überprüfen Sie, ob ein Rückverweis zum GC-Stamm vorhanden ist.

Verwaiste Objekte suchen

1. Erstellen Sie zwei Speicherabbilder.

   Siehe „Speicherabbild erstellen und anzeigen“ auf Seite 152.

2. Wählen Sie die zwei Speicherabbilder für den Vergleich aus.

   Hinweis: *Es kann kein drittes Abbild herangezogen werden. gleichzeitig ist immer nur ein Vergleich von zwei Speicherabbildern möglich.*

3. Klicken Sie auf die Schaltfläche „Verwaiste Objekte suchen“.

Die Verwaiste-Objekte-Ansicht zeigt Ihnen potenzielle Speicherlecks. Die Spalten „Instanzen“ und „Speicher“ zeigen die Unterschiede zwischen der Anzahl Instanzen einer Klasse und der von ihnen belegten Speichermenge während des Intervalls zwischen zwei Speicherabbildern. Wenn Sie eine Klasse sehen, die während dieses Zeitraums nicht hätte erstellt werden sollen, bzw. eine Klasse, die gelöscht werden sollte, überprüfen Sie die Anwendung, um zu ermitteln, ob dies die Ursache eines Speicherlecks ist.

4 Um zu bestimmen, wie ein Objekt in der Ansicht „Verwaiste Objekte suchen“ instanziiert wurde, doppelklicken Sie in der Ansicht auf das entsprechende Objekt. Die Objektverweiseansicht zeigt das Stacktrace für jede ausgewählte Instanz an.

Eine Methode zum Identifizieren eines Speicherlecks besteht darin, zunächst eine bestimmte Schrittfolge wiederholt auszuführen, wodurch immer mehr Speicher belegt wird. Es ist wichtig, diese Schrittfolge wenigstens einmal in der Anwendung auszuführen, bevor Sie das anfängliche Speicherabbild erstellen, um auf diese Weise im Zwischenspeicher abgelegte Objekte oder andere Instanzen im Abbild einzuschließen.

Anschließend wiederholen Sie diese Schrittfolge eine bestimmte Anzahl von Malen (3, 7 oder eine andere Primzahl) in der Anwendung. Erstellen Sie dann das zweite Speicherabbild, um es mit dem ersten zu vergleichen. In der Ansicht „Verwaiste Objekte suchen“ finden Sie dann möglicherweise verwaiste Objekte mit einem Mehrfachen von 3 oder 7 Instanzen. Diese Objekte sind wahrscheinlich Speicherlecks. Doppelklicken Sie auf die Klassen, um sich die Stracktraces für jede Instanz anzusehen.

Eine andere Methode besteht darin, die Schrittfolge über einen längeren Zeitraum zu wiederholen und zu warten, bis die Speicherauslastung ihr Maximum erreicht. Wenn die Speicherauslastung danach nicht höher wird, enthält diese Schrittfolge kein Speicherleck.

Häufige Ursachen von Speicherlecks sind auch verwaiste Ereignis-Listener. Sie können die Methode `removeEventListener()` verwenden, um nicht mehr benötigte Ereignis-Listener zu entfernen. Weitere Informationen finden Sie unter Objekterstellung und -entfernung in Verwenden von Flex 4.6.

Ausführungszeiten analysieren

Anhand der Analyse der Ausführungszeiten von Methoden und der Menge des zugewiesenen Speicherplatzes können Sie feststellen, wo Leistungsengpässe auftreten.

Dies ist besonders hilfreich, wenn Sie die Ausführungszeit von Methoden ermitteln, die viele Male aufgerufen werden.

Häufigkeit von Methodenaufrufen ermitteln

1 Starten Sie eine Profilingsitzung und aktivieren Sie im Startbildschirm beim Konfigurieren des Profilers die Option „Leistungs-Profiling“.

2 Wählen Sie die Anwendung in der Profilansicht aus.

3 Interagieren Sie mit der Anwendung, bis Sie zu dem Punkt gelangen, wo Sie mit der Analyse der Anzahl der Methodenaufrufe beginnen möchten. Um zu sehen, wie oft eine Methode seit dem Starten der Anwendung aufgerufen wurde, dürfen Sie nicht mit der Anwendung interagieren.

4 Klicken Sie auf die Schaltfläche „Leistungsdaten zurücksetzen“. Hierdurch werden sämtliche Leistungsdaten gelöscht, damit das nächste Leistungsprofil nur Daten von diesem Punkt an enthält.

5 Interagieren Sie mit der Anwendung, bis Sie an den Punkt gelangen, wo Sie die Anzahl der Methodenaufrufe seit dem Rücksetzen der Leistungsdaten überprüfen.

6 Klicken Sie auf die Schaltfläche „Leistungsprofil erfassen“.

7 Doppelklicken Sie in der Profilansicht auf das Leistungsprofil.

8 Sortieren Sie die Daten in der Leistungsprofilansicht nach der Spalte „Methode“ und suchen Sie Ihre Methode in der Liste.

Der Wert in der Spalte „Aufrufe“ zeigt an, wie oft diese Methode im Samplingintervall aufgerufen wurde. Hierbei handelt es sich um den Zeitraum zwischen dem Klick auf „Leistungsdaten zurücksetzen“ und „Leistungsprofil erfassen“.

Prüfen Sie die Werte unter „Kumulative Zeit“, „Eigene Zeit“, „Durchschnittliche kumulative Zeit“ und „Durchschnittliche eigene Zeit“ in der Leistungsprofilansicht. Diese geben die Ausführungszeit der Methoden an.

Vergleichen Sie die für die Ausführung der einzelnen Methoden benötigte Zeit mit der Zeit, die von allen von einer bestimmten Methode aufgerufenen Methoden benötigt wird. Im Allgemeinen gilt: Wenn der Wert für die eigene Zeit bzw. durchschnittliche eigene Zeit einer Methode im Vergleich zu anderen Methoden hoch ist, untersuchen Sie, wie die Methode implementiert wird und versuchen Sie, die Ausführungszeit zu reduzieren.

Genauso gilt aber auch: Wenn der Wert für die eigene Zeit bzw. durchschnittliche eigene Zeit einer Methode niedrig ist, aber die kumulative Zeit oder der Durchschnitt der kumulativen Zeit hoch ist, prüfen Sie die Methoden, die von dieser Methode aufgerufen werden, um den Engpass zu finden.

Übermäßige Objektzuweisungen finden

Eine Möglichkeit zur Identifizierung von Problembereichen in einer Anwendung ist die Suche nach Quellen, die die Erstellung einer übermäßig hohen Anzahl von Objekten verursachen. Das Erstellen einer Objektinstanz kann ressourcenintensiv sein, besonders, wenn dieses Objekt in der Anzeigeliste steht. Das Hinzufügen eines Objekts zur Anzeigeliste kann viele Aufrufe von Stil- und Layoutmethoden verursachen. Dies verlangsamt die Anwendung möglicherweise. In einigen Fällen können Sie Code refaktorieren, damit weniger Objekte erstellt werden.

Nachdem Sie ermittelt haben, ob Objekte unnötig erstellt werden, finden Sie heraus, ob es sich lohnt, die Anzahl der Instanzen dieser Klasse zu reduzieren. Beispiel: Finden Sie heraus, wie groß die Objekte sind. Denn größere Objekte haben im Allgemeinen das höchste Optimierungspotenzial.

Um festzustellen, welche Objekte in großer Anzahl erstellt werden, vergleichen Sie Speicherabbilder von zwei verschiedenen Zeitpunkten der Anwendung.

Anzahl von Instanzen einer bestimmten Klasse anzeigen

1 Starten Sie eine Profilingsitzung und aktivieren Sie Speicher-Profiling, wenn Sie den Profiler auf der Startseite konfigurieren.

2 Interagieren Sie mit der Anwendung, bis Sie an die Stelle gelangen, wo das Speicherabbild erstellt werden soll.

3 Klicken Sie auf die Schaltfläche „Speicherabbild erstellen“. Der Profiler fügt ein neues Speicherabbild zur Anwendungsliste der Profilansicht hinzu.

4 Öffnen Sie das Speicherabbild, indem Sie in der Profilansicht darauf doppelklicken.

5 Um die Instanzenanzahl einer bestimmten Klasse sowie den Speicherbedarf dieser Instanzen anzuzeigen, sortieren Sie nach der Spalte „Klasse“ und suchen Sie Ihre Klasse in dieser Spalte. Sie können auch nach den anderen Spalten sortieren, um die Objekte mit dem höchsten Speicherbedarf oder den meisten Instanzen zu identifizieren. In den meisten Fällen sind Strings die Klasse mit den meisten Instanzen und dem größten Speicherbedarf.

Weitere Informationen zur Speicherabbildansicht finden Sie unter „Speicherabbildansicht“ auf Seite 151.

Instanzen übermäßiger Objektzuweisung auffinden

1 Starten Sie eine Profilingsitzung und aktivieren Sie Speicher-Profiling, wenn Sie den Profiler auf der Startseite konfigurieren.

2 Interagieren Sie mit der Anwendung, bis Sie an die erste Stelle gelangen, an der ein Speicherabbild erstellt werden soll.

3 Klicken Sie auf die Schaltfläche „Speicherabbild erstellen“.

Der Profiler speichert das Speicherabbild in der Profilansicht und versieht es mit einem Zeitstempel.

4 Interagieren Sie mit der Anwendung, bis Sie an die zweite Stelle gelangen, an der ein Speicherabbild erstellt werden soll.

5 Klicken Sie erneut auf die Schaltfläche „Speicherabbild erstellen“.

Der Profiler speichert das zweite Speicherabbild in der Profilansicht und versieht es mit einem Zeitstempel.

6 Wählen Sie die zwei Speicherabbilder für den Vergleich aus.

Hinweis: *Es kann kein drittes Abbild herangezogen werden. Sie können jeweils immer nur zwei Abbilder vergleichen.*

7 Klicken Sie auf die Schaltfläche „Speicherreservierung einblenden“.

Die Speicherreservierungsansicht zeigt, welche Methoden zwischen den zwei Abbildern aufgerufen wurden und wie viel Speicherplatz während dieser Aufrufe beansprucht wurde. Siehe „Speicherreservierungsansicht“ auf Seite 155.

Profiler-Filter

Die in der Profileransicht angezeigte Datenmenge kann manchmal überwältigend und zu detailliert sein. Die internen Aktionen von Flash Player verdecken eventuell die Daten, an denen Sie tatsächlich interessiert sind, beispielsweise eigene Methoden und Klassen. Außerdem erstellt und zerstört Flash Player viele Objekte ohne Ihr direktes Eingreifen. Daher sehen Sie wahrscheinlich, dass in den Anwendungen Tausende von Strings oder Arrays verwendet werden.

In den folgenden Ansichten können Sie Filter verwenden:

- Live-Objekte
- Speicherabbild
- Leistungsprofil
- Methodenstatistiken
- Speicherreservierung

Sie können unter Verwendung der Profilerfilter definieren, welche Pakete in den Profileransichten angezeigt werden sollen. Es gibt zwei Arten von Filtern:

Ausschlussfilter Die Ausschlussfilter weisen den Profiler an, Pakete, deren Muster mit denen der Musterliste übereinstimmen, aus den Profileransichten auszuschließen. Wenn Sie beispielsweise Diagrammsteuerelemente verwenden, sie aber nicht im Profil haben möchten, können Sie das Muster mx.charts.* zum Ausschlussfilter hinzufügen. Sie können auch globale integrierte Elemente ausschließen. Hierzu gehören globale Klassen wie String und Array.

Einschlussfilter Die Einschlussfilter weisen den Profiler an, nur Pakete, deren Muster mit denen der Musterliste übereinstimmen, in die Profileransichten einzuschließen. Wenn Sie beispielsweise ein benutzerdefiniertes Paket namens com.mycompany.* haben, können Sie Informationen ausschließlich über die Klassen dieses Pakets anzeigen, indem Sie es zum Einschlussfilter hinzufügen.

Die Standardausschlüsse sind flash.*.*, spark.*.*, mx.*.* und die Flex Framework-Klassen im globalen oder unbenannten Paket. Hierzu gehören globale Klassen wie String und Array. Die Standardeinschlüsse sind also benutzerdefinierte Klassen im unbenannten Paket und benutzerdefinierte Klassen in nicht zum Framework gehörigen Paketen (wie z. B. com.mycompany.MyClass).

Sie können benutzerdefinierte Klassen, die sich im unbenannten Paket befinden, von den Profilingdaten ausschließen. Fügen Sie hierzu „*“ zur Ausschlussliste hinzu.

Maximal sichtbare Zeilen Mit der Option „Maximal sichtbare Zeilen“ wird angegeben, wie viele Datenreihen in einer Ansicht angezeigt werden können. Erhöhen Sie diesen Wert, wenn die Daten, nach denen Sie suchen, nicht in der Ansicht angezeigt werden. Verringern Sie diesen Wert, um die Leistung des Profilers zu verbessern. Verwenden Sie andere Filter, um die Daten, die Sie interessieren, zu sehen.

Maximal zurückgegebene Anzahl Rückverweispfade: Mit der Option „Maximal zurückgegebene Anzahl Rückverweispfade“ wird festgelegt, wie viele Pfade zu einem verwiesenen Objekt beim Überprüfen von Objektverweisen angezeigt werden. Die Pfade werden entsprechend dem kürzesten Pfad angezeigt. In der Standardeinstellung werden die 10 kürzesten Pfade angezeigt. Erhöhen Sie diesen Wert, um zusätzliche Pfade anzuzeigen, oder aktivieren Sie die Option „Alle Rückverweispfade anzeigen“. Wenn Sie zusätzliche Pfade anzeigen, können Sie Objekte finden, auf die nicht mehr verwisen wird. Siehe „Speicherlecks finden“ auf Seite 173.

Standardvoreinstellungen für Filter festlegen

❖ Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Profiler“ > „Einschlussfilter“ bzw. „Ausschlussfilter“.

Beim Anzeigen von Profilerdaten wendet der Profiler zuerst die Ausschlussfilter und dann die Einschlussfilter an. Ein Beispiel: Sie setzen den Ausschlussfilter auf mx.controls.*, aber den Einschlussfilter auf mx.*.*. Der Profiler zeigt keine Informationen zu Klassen im mx.controls-Paket an, da das Paket ausgeschlossen war (und obwohl ihr Muster mit dem Muster der Einschlussliste übereinstimmt). Ein anderes Beispiel: Sie setzen den Ausschlussfilter auf mx.*.* und den Einschlussfilter auf mx.controls.*. Der Profiler zeigt keine Informationen zu Klassen im Paket mx.controls.* an, da sie ausgeschlossen wurden, bevor sie eingeschlossen wurden.

Wenn Sie bestimmte Datenpunkte herausfiltern, werden die Prozentwerte der Spalten angepasst, um nur den Prozentsatz der nicht gefilterten Daten wiederzugeben.

Der Profiler behält die Filter von einer Profilingsitzung bis zur nächsten Sitzung für dieselbe Anwendung bei.

Die Filtereinstellungen werden bei Unteransichten nicht vererbt. Beispiel: Wenn Sie einen Filter auf die Daten in der Speicherabbildansicht anwenden und dann zur Objektverweiseansicht wechseln, indem Sie auf eine Methode doppelklicken, wendet die Objektverweiseansicht nicht denselben Filter an.

Ermitteln, ob Daten gefiltert werden

1 Klicken Sie auf die Schaltfläche „Filter“ oder sehen Sie sich die Titel der Datentabellen an. Wenn Filter angewendet wurden, lautet die Überschrift der Paketspalte „Paket (gefiltert)“.

2 (Optional) Setzen Sie die Filter auf die Standardeinstellungen zurück, indem Sie auf die Schaltfläche „Standardwerte wiederherstellen“ klicken.

Kapitel 7: Unit-Test-Tools in Flash Builder

FlexUnit-Testumgebung

In der FlexUnit-Testumgebung können Sie wiederholbare Tests generieren und bearbeiten, die von Skripten oder direkt von Flash Builder aus ausgeführt werden können. Flash Builder unterstützt die Open Source-Frameworks FlexUnit 4 und Flex Unit 1.

Von Flash Builder aus können Sie Folgendes tun:

- Erstellen von Unit-TestCases und Unit-TestSuites

  Assistenten in Flash Builder führen Sie durch die Erstellung von TestCases und TestSuites und generieren die Codeabschnitte für die Tests.

- Ausführen von TestCases und TestSuites

  Sie können TestCases und TestSuites auf verschiedene Weisen von Flash Builder bzw. außerhalb der Flash Builder-Umgebung ausführen. Die Testergebnisse werden in einer Testanwendung angezeigt. Flash Builder öffnet eine FlexUnit-Ergebnisansicht für die Analyse des Testlaufs.

- Navigieren von der FlexUnit-Ergebnisansicht zum Quellcode

  Doppelklicken Sie im Testergebnisbereich auf einen Test, um die Testimplementierung zu öffnen.

  Im Bereich „Tests mit Fehlern - Details“ sind die Quelle und die Zeilennummer des Fehlers aufgelistet. Wenn die aufgelistete Quelle im aktuellen Arbeitsbereich ist, doppelklicken Sie auf die Quelle, um direkt zu ihr zu gehen.

Die Grundlagen der testgesteuerten Entwicklung unter Verwendung von FlexUnit und Flash Builder werden in diesem englischsprachigen Artikel im Adobe Developer Center vom Adobe Community-Experten Elad Elrom erklärt.

FlexUnit-Tests erstellen

FlexUnit-TestCase-Klassen und TestCase-Suites können für die folgenden Arten von Projekten erstellt werden:

- Flex-Projekte
- Flex-Mobilprojekt
- Flex-Bibliotheksprojekte
- ActionScript-Projekte
- AIR-Projekte

Wenn Sie eine TestCase-Klasse erstellen, können Sie die folgenden Optionen festlegen:

- Einen `src`-Ordner in einem Flex-Projekt für die Klasse
- Ein Paket für die Klasse
- Die zu testenden Klassen
- Methoden, die für jede angegebene Klasse getestet werden sollen

Eine FlexUnit-TestCase-Suite ist eine Reihe von Tests, die auf zuvor erstellten TestCase-Klassen, in diesen Klassen angegebenen Methoden und anderen TestCase-Suites basieren.

FlexUnit-TestCase-Klassen erstellen

Wenn Sie eine FlexUnit-TestCase-Klasse erstellen, generiert Flash Builder eine ActionScript-Datei für die TestCase-Klasse, die in einem Paket für TestCases abgelegt wird.

Bei den folgenden Schritten wird angenommen, dass Sie ein Projekt in Flash Builder erstellt haben, in dem Sie FlexUnit-Tests erstellen und ausführen möchten.

1 Wählen Sie das Flex-Projekt aus und wählen Sie dann im Kontextmenü „Neu“ > „TestCase-Klasse“.

Wenn Sie im Projekt eine ActionScript-Klassendatei wählen, wird diese Klasse automatisch im Assistenten „Neue Testfallklasse“ für den FlexUnit-TestCase gewählt.

2 Geben Sie im Assistenten „Neue Testfallklasse“ an, ob eine Klasse im FlexUnit 4-Stil oder im FlexUnit 1-Stil erstellt werden soll.

3 Geben Sie einen Namen für die TestCase-Klasse ein.

4 (Optional) Geben Sie einen Quellordner und ein Paket für die TestCase-Klasse an oder übernehmen Sie die Standardeinstellungen.

Der Standardquellordner ist der Ordner `src` des aktuellen Projekts. Das Standardpaket ist `flexUnitTests`. Er befindet sich in der obersten Ebene der standardmäßigen Paketstruktur des Projekts.

5 (Optional) Aktivieren Sie die Option „Zu testende Klasse auswählen“ und wählen Sie dann eine Klasse. Klicken Sie auf „Weiter“.

6 (Optional) Wählen Sie die Methoden in der ausgewählten Klasse aus, die Sie testen möchten.

7 Klicken Sie auf „Fertig stellen“.

Kodieren Sie den von Ihnen erstellten TestCase. Verwenden Sie die generierten Code-Stubs als Ausgangspunkt.

FlexUnit-TestCase-Suites erstellen

Bei den folgenden Schritten wird angenommen, dass Sie bereits TestCase-Klassen erstellt haben.

1 Wählen Sie das Flex-Projekt aus und erstellen Sie mithilfe der Kontextmenüs eine TestCase-Suite, indem Sie „Neu“ > „TestSuite-Klasse“ wählen.

2 Geben Sie im Assistenten „Neue Testsuite-Klasse“ an, ob eine Klasse im FlexUnit 4-Stil oder im FlexUnit 1-Stil erstellt werden soll.

3 Geben Sie einen Namen für die Testsuite ein.

4 Navigieren Sie in den TestSuites und TestCases, um die Klassen und Methoden auszuwählen, die in die TestSuite aufgenommen werden sollen. Klicken Sie auf „Fertig stellen“.

Standardmäßige FlexUnit-TestCase-Klassen und -TestSuite-Klassen anpassen

Sie können die von Flash Builder erstellten standardmäßigen FlexUnit-TestCase-Klassen und TestSuite-Klassen anpassen. In Flash Builder werden die Standardversionen dieser Dateien mithilfe von Dateivorlagen erstellt.

Sie können die Dateivorlagen von FlexUnit im Dialogfeld „Voreinstellungen“ aufrufen. Wählen Sie dazu die Optionen „Flash Builder“ > „Dateivorlagen“ > „FlexUnit“. Für TestCase-Klassen und TestSuite-Klassen von FlexUnit1 und FlexUnit4 stehen separate Vorlagen zur Verfügung.

Weitere Informationen zum Ändern der standardmäßigen Dateivorlagen finden Sie unter „Dateivorlagen anpassen“ auf Seite 63.

Hinweis: *`FlexUnitCompilerApplication.mxml` und `FlexUnitApplication.mxml` sind von der Vorlage für MXML-Webanwendung oder MXML-Desktopanwendung abgeleitet. Welche Vorlage verwendet wird, hängt davon ab, ob das Flex-Projekt für eine Webanwendung (wird in Adobe® Flash® Player ausgeführt) oder Desktopanwendung (wird in Adobe AIR® ausgeführt) konfiguriert ist.*

Verwandte Themen

Open Source-Sprachreferenz zu FlexUnit

Open Source-Dokumentation zu FlexUnit

FlexUnit-Tests ausführen

FlexUnit-Tests können direkt in Flash Builder oder außerhalb von Flash Builder mit SWF-Dateien ausgeführt werden, die für den FlexUnit-Test generiert wurden. In beiden Fällen werden die Ergebnisse der Tests in der FlexUnit-Ergebnisansicht angezeigt.

Sie können FlexUnit-Tests aber auch konfigurieren und speichern und anschließend ausführen.

Standardmäßig werden FlexUnit-Tests in der Flash-Debugperspektive ausgeführt. Sie können Tests zwar in der Flash-Entwicklungsperspektive oder der Flash-Profilperspektive ausführen, aber Flash Builder wechselt beim Ausführen des Tests in die Flash-Debugperspektive.

Sie können die Standardperspektive für FlexUnit-Tests ändern. Öffnen Sie das Dialogfenster „Voreinstellungen“ und wählen Sie „Flash Builder“ > „FlexUnit“.

FlexUnit-Compiler-Anwendung und FlexUnit-Anwendung

Wenn Sie einen FlexUnit-TestCase erstellen, erstellt Flash Builder die folgende FlexUnit-Compiler-Anwendung und eine FlexUnit-Anwendung:

- FlexUnitCompilerApplication.mxml
- FlexUnitApplication.mxml

Flash Builder nutzt diese Anwendungen beim Kompilieren und Ausführen von FlexUnit-Tests. Flash Builder platziert die Anwendungen im `src`-Ordner des Projekts.

Diese Anwendung enthält Verweise auf alle von Flash Builder generierten FlexUnit-TestCases und TestSuites. Flash Builder platziert alle FlexUnit-Tests in das `<fx:Declarations>`-Tag dieser Anwendung. Normalerweise bearbeiten oder ändern Sie diese Datei nicht direkt.

Aktualisieren Sie die FlexUnit-Compiler-Anwendung in den folgenden Situationen:

- Sie fügen einen TestCase manuell hinzu.

  Wenn Sie eine TestCase-Klasse ohne den Assistenten „Neue Testfallklasse“ erstellen, aktualisieren Sie `FlexUnitCompilerApplication.mxml`. Platzieren Sie den neuen TestCase im Paket mit den anderen TestCases.
- Sie benennen einen TestCase um.
- Sie löschen einen TestCase.

Aktualisieren Sie `FlexUnitCompilerApplication.mxml`:

1 Wenn die Ansicht „FlexUnit-Ergebnisse“ nicht angezeigt wird, wählen Sie „Fenster“ > „Ansicht anzeigen“ > „Andere“ > Flash Builder“ > „FlexUnit-Ergebnisse“. Klicken Sie auf „OK“.

2 Klicken Sie in der Ansicht „FlexUnit-Ergebnisse“ auf die Schaltfläche zum Aktualisieren.

FlexUnit-Tests in Flash Builder ausführen

In Flash Builder können Sie FlexUnit-Tests auf Projektebene oder für einzelne TestCases ausführen.

Normalerweise führen Sie FlexUnit-Tests über das Kontextmenü eines Projekts oder die Kontextmenüs eines einzelnen TestCase aus.

Sie können FlexUnit-Tests jedoch auch über das Flash Builder-Menü „Ausführen“, das Flash Builder-Symbol für „Ausführen“ oder mithilfe des Symbols für „FlexUnit-Tests ausführen“ in der Ansicht „FlexUnit-Ergebnisse“ ausführen.

Wenn Sie Tests im Flash Builder-Menü „Ausführen“ ausführen, wird ein Dialogfenster zum Konfigurieren des Tests geöffnet. Hier können Sie festlegen, welche Testklassen und Methoden ausgeführt werden sollen. TestCases für Bibliotheksprojekte können nicht über das Flash Builder-Menü „Ausführen“ ausgeführt werden.

In Flash Builder können Sie mithilfe der folgenden Tastaturbefehle schnell FlexUnit-Tests ausführen:

- `Alt+Umschalt+A, F`

  Führt alle FlexUnit-Tests im Projekt aus.

- `Alt+Umschalt+E, F`

  Führt den gewählten FlexUnit-Test aus.

Führen Sie FlexUnit-Tests für die aktuelle Auswahl im Editor aus. Siehe „FlexUnit-Tests konfigurieren“ auf Seite 182.

1 Wählen Sie ein Projekt und führen Sie den Test folgendermaßen aus:

  Wählen Sie im Kontextmenü eines Projekts „FlexUnit-Tests ausführen“.

  Wählen Sie im Flash Builder-Menü unter „Ausführen“ oder über das Symbol für „Ausführen“ den Befehl „FlexUnit-Tests“.

2 (Flash Builder-Menü „Ausführen“) Wählen Sie im Konfigurationsdialogfeld „FlexUnit-Tests ausführen“ die TestCases und Methoden aus, die im Test ausgeführt werden sollen. Klicken Sie auf „OK“, um die Tests auszuführen.

3 Zeigen Sie die Testergebnisse an.

  Flash Builder generiert eine SWF-Datei im Ordner „bin-debug“ des Projekts.

  Es wird eine Anwendung geöffnet, in der Informationen zum Test sowie zum Status des Tests (d. h. es wird angezeigt, wenn der Test abgeschlossen ist) angezeigt werden.

  Das Bedienfeld mit der FlexUnit-Ergebnisansicht wird geöffnet und zeigt die Ergebnisse des Tests an. Siehe „Ergebnisse eines FlexUnit-Testlaufs anzeigen“ auf Seite 183.

Ausführen von einzelnen FlexUnit-Tests:

1 Navigieren Sie im Projektexplorer folgendermaßen zum Paket `flexUnitTest`:

  Wählen Sie im Kontextmenü einer FlexUnit-Testdatei „FlexUnit-Tests ausführen“.

2 Zeigen Sie die Testergebnisse an.

Flash Builder generiert eine SWF-Datei im Ordner „bin-debug“ des Projekts.

Es wird eine Anwendung geöffnet, in der Informationen zum Test sowie zum Status des Tests (d. h. es wird angezeigt, wenn der Test abgeschlossen ist) angezeigt werden.

Das Bedienfeld mit der FlexUnit-Ergebnisansicht wird geöffnet und zeigt die Ergebnisse des Tests an. Siehe „Ergebnisse eines FlexUnit-Testlaufs anzeigen“ auf Seite 183.

FlexUnit-Tests außerhalb der Flash Builder-Umgebung ausführen

Bei den folgenden Schritten wird angenommen, dass Sie bereits einen FlexUnit-Test in FlashBuilder ausgeführt haben und dass Flash Builder momentan ausgeführt wird.

1 Kopieren Sie die für einen Test generierte SWF-Datei aus dem Ordner „bin-debug“ in Ihrem Projekt in einen Ordner außerhalb Ihrer Entwicklungsumgebung.

Sie können die automatisch generierte SWF-Datei oder eine SWF-Datei aus einem FlexUnit-Test, den Sie bereits konfiguriert und gespeichert haben, kopieren.

2 Führen Sie die Kopie der SWF-Datei aus.

Flash Player wird geöffnet und zeigt Informationen zum Test sowie zum Status des Tests (d. h. es wird angezeigt, wenn der Test abgeschlossen ist) an.

Das Bedienfeld mit der FlexUnit-Ergebnisansicht wird in Flash Builder geöffnet und zeigt die Ergebnisse des Tests an.

FlexUnit-Tests konfigurieren

1 Öffnen des Konfigurationsdialogfelds „FlexUnit-Tests ausführen“:

Sie können das Konfigurationsdialogfeld „FlexUnit-Tests ausführen“ über das Menü „Ausführen“ oder die Ansicht „FlexUnit-Ergebnisse“ aufrufen.

- Wählen Sie ein Projekt aus. Wählen Sie im Flash Builder-Menü „Ausführen“ > „Ausführen“ > „FlexUnit-Tests“.
- Klicken Sie in der Ansicht „FlexUnit-Ergebnisse“ auf das Symbol für „FlexUnit-Tests ausführen“.

  Wenn die Ansicht „FlexUnit-Ergebnisse“ nicht angezeigt wird, wählen Sie „Fenster“ > „Ansicht anzeigen“ > „Andere“ > Flash Builder“ > „FlexUnit-Ergebnisse“.

2 Wählen Sie im Dialogfeld „Testkonfiguration“ die TestCases und Methoden aus, die als Testkonfiguration gespeichert werden sollen.

Hinweis: *Das Dialogfeld „Testkonfiguration“ ist nicht verfügbar, wenn Sie Tests über das Kontextmenü im Paket-Explorer ausführen.*

3 (Optional) Wählen Sie „Laden“, um vorherige Konfigurationen, die als XML-Dateien gespeichert wurden, zu importieren.

4 Klicken Sie auf „Speichern“.

Flash Builder schreibt eine XML- und eine MXML-Datei in den zum Projekt gehörenden Ordner `.FlexUnitSettings`.

Sie können die XML-Datei in Erstellungsskripten verwenden, um den Test auszuführen.

Sie können aus der MXML-Datei eine SWF-Datei generieren. Diese SWF-Datei kann für Tests außerhalb der Flash Builder-Umgebung verwendet werden. Normalerweise kopieren Sie zum Generieren der SWF-Datei die generierte MXML-Datei in den `src`-Ordner Ihres Projekts.

Ergebnisse eines FlexUnit-Testlaufs anzeigen

Die FlexUnit-Ergebnisansicht zeigt Ergebnisse von FlexUnit-Tests sowie Informationen zu fehlgeschlagenen Tests an. Sie können durch die Ergebnisse blättern, die Anzeige filtern, die Ergebnisse in eine Datei schreiben bzw. Ergebnisse aus einer Datei laden.

Sie können auch Tests erneut ausführen, einen Test abbrechen, der gerade ausgeführt wird, und die Ergebnisse aus der Ansicht löschen.

Wenn die Ansicht „FlexUnit-Ergebnisse“ nicht angezeigt wird, wählen Sie „Fenster“ > „Ansicht anzeigen“ > „Andere“ > Flash Builder“ > „FlexUnit-Ergebnisse“.

Testergebnisbereich

Dieser Bereich listet alle Tests im Testlauf auf und zeigt den Status des Tests (Bestanden, nicht bestanden) an.

Doppelklicken Sie auf einen Test in der Liste, um den jeweiligen Test im ActionScript-Editor anzuzeigen.

Bereich „Tests mit Fehlern - Details“

Wählen Sie im Testergebnisbereich einen Test aus, um Details anzuzeigen, warum der Test nicht bestanden wurde.

Im Bereich werden die Quelldatei und Methode sowie die Zeilennummer des Fehlers angezeigt.

Wenn die Quelldatei eine lokale Datei im Arbeitsbereich ist, können Sie auf die Datei in der Liste doppelklicken, um den Fehler im ActionScript-Editor anzuzeigen.

FlexUnit-Ergebnisse (Ansichtsmenü)

Im FlexUnit-Ergebnisbereich (über das Ansichtsmenü aufrufbar) können Sie Folgendes tun:

- Filterung der angezeigten Ergebnisse

  Ausblenden des Bereichs, in dem die Fehlerdetails angezeigt werden.

  Anzeige der Testläufe, die fehlgeschlagen sind.
- Durchblättern der Testläufe, die im Testergebnisbereich angezeigt werden.
- Abbrechen eines laufenden Tests.
- Speichern der Ergebnisse eines Testlaufs oder der Konfiguration eines Testlaufs.
- Laden von zuvor in einer Datei gespeicherten Ergebnissen.
- Löschen der Ergebnisse aus dem Bereich.
- Erneutes Ausführen des aktuellen Tests Zur Auswahl stehen folgende Optionen:
  - Alle Tests ausführen
  - Nur Tests mit Fehlern ausführen.
  - Ausgewählten Test ausführen.

- Aktualisieren der FlexUnit-Konfiguration.

  Wenn Sie einen Test modifizieren oder Tests hinzufügen oder entfernen, klicken Sie auf das Symbol für „Aktualisieren“, um die neue FlexUnit-Konfiguration zu laden.

- Konfigurieren und Ausführen von FlexUnit-Tests.

  Klicken Sie auf das Symbol für „FlexUnit-Tests ausführen“, um FlexUnit-Tests zu konfigurieren und auszuführen.

FlexUnit-Unterstützung für Mobilprojekte

Sie können einen FlexUnit-Test auf einem Mobilgerät ausführen. Sollten Sie über kein Mobilgerät verfügen, können Sie das Gerät auf Ihrem Desktop simulieren und den FlexUnit-Test dort ausführen.

FlexUnit auf einem Mobilgerät ausführen

1 Wählen Sie ein Mobilprojekt und führen Sie den Test aus, indem Sie „Ausführen“ > „FlexUnit-Tests“ oder die Option zum Ausführen von FlexUnit-Tests über das Kontextmenü wählen.

2 Geben Sie eine Startmethode an:

   - **Auf Desktop** Führt die FlexUnit-Anwendung auf Ihrem Desktop mit dem AIR Debug Launcher (ADL) gemäß einer angegebenen Gerätekonfiguration aus.
   - **Auf Gerät** Führt die FlexUnit-Anwendung auf dem Gerät aus

     Flash Builder kann auf das Gerät über den USB-Port Ihres Computers oder über das Netzwerk per Wi-Fi zugreifen. Eine Socketverbindung und die IP-Adresse des Computers werden für die Kommunikation zwischen Flash Builder und der auf dem Gerät ausgeführten FlexUnit-Anwendung verwendet.

     Hinweis: *Sie können den FlexUnit-Test auf einem Gerät, das über den USB-Port Ihres Computers verbunden ist, ausführen. Sie können die Testergebnisse jedoch nur anzeigen, wenn Wi-Fi auf dem Gerät aktiviert ist.*

     Die Standardportnummer, die zum Herstellen einer Socketverbindung verwendet wird, lautet 8765. Sie können die Standardportnummer ändern, indem Sie zum Dialogfeld „Voreinstellungen“ navigieren und „Flash Builder“ > „FlexUnit“ auswählen.

     Kann die FlexUnit-Anwendung keine Verbindung mit dem Gerät herstellen, zeigt Flash Builder manchmal ein Dialogfeld an, in dem die IP-Adresse des Computers angefordert wird. Stellen Sie sicher, dass das Gerät ordnungsgemäß mit dem Netzwerk verbunden ist und auf den Computer, auf dem Flash Builder ausgeführt wird, von diesem Netzwerk aus zugegriffen werden kann.

3 Wählen Sie im Konfigurationsdialogfeld „FlexUnit-Tests ausführen“ die TestCases und Methoden aus, die im Test ausgeführt werden sollen. Klicken Sie auf „OK“, um die Tests auszuführen.

4 Zeigen Sie die Testergebnisse an.

   Flash Builder generiert eine SWF-Datei im Ordner „bin-debug“ des Projekts.

   Es wird eine Anwendung geöffnet, in der Informationen zum Test sowie zum Status des Tests (d. h. es wird angezeigt, wenn der Test abgeschlossen ist) angezeigt werden.

   Das Bedienfeld mit der FlexUnit-Ergebnisansicht wird geöffnet und zeigt die Ergebnisse des Tests an. Siehe „Ergebnisse eines FlexUnit-Testlaufs anzeigen“ auf Seite 183.

Kapitel 8: Entwickeln von Anwendungen mit Flex

Einfacher Arbeitsablauf der Anwendungsentwicklung mit Flex

1 (Optional) Erstellen oder importieren Sie ein Projekt.

Zum Erstellen eines Projekts wählen Sie in Flash Builder „Datei“ > „Neu“ und dann den Projekttyp, den Sie öffnen möchten. Der Projekterstellungsassistent ist für jedes Projekt verschieden.

In einigen Fällen greift das Projekt auf Serverdaten zu, wie z. B. von einem PHP-, ColdFusion- oder Java-Anwendungsserver. Geben Sie beim Erstellen des Projekts Informationen für den Zugriff auf den Server an. Weitere Informationen finden Sie unter „Projekte in Flash Builder erstellen“ auf Seite 90.

Sie können Projekte importieren, die zuvor in Flash Builder erstellt oder daraus exportiert wurden. Weitere Informationen finden Sie unter „Projekte exportieren und importieren“ auf Seite 97.

Außerdem können Sie Projekte importieren, die in Adobe Flash Catalyst erstellt wurden. Weitere Informationen finden Sie unter „Verwenden von Flash Builder mit Flash Catalyst“ auf Seite 257.

2 Erstellen oder bearbeiten Sie die Quelldateien eines Projekts.

Mithilfe der Flash Builder-Editoren können Sie Quelldateien eines Projekts bearbeiten. In Flash Builder gibt es den Quellmodus und den Designmodus. Viele Entwickler gestalten Benutzeroberflächenelemente im Designmodus und geben Code im Quellmodus ein.

Weitere Informationen finden Sie unter „Tools zur Codeentwicklung in Flash Builder“ auf Seite 49.

3 Erstellen Sie das Projekt.

Standardmäßig erstellt Flash Builder jedes Mal ein Projekt, wenn Sie eine Datei in dem Projekt speichern. Beim Erstellen des Projekts platziert Flash Builder die Ausgabedateien in den entsprechenden Speicherort für das Projekt. Flash Builder informiert Sie über Fehler oder Warnungen, die beim Erstellen des Projekts aufgetreten sein können.

Weitere Informationen finden Sie unter „Projekte erstellen“ auf Seite 103.

4 Führen Sie Anwendungen in dem Projekt aus.

Jedes Projekt (außer Bibliotheksprojekte) kann Anwendungsdateien enthalten, die Sie in Ausführungs-, Debug- oder Profilperspektiven von Flash Builder starten können. Flash Builder definiert eine Startperspektive zum Starten der Anwendung. Sie können die Standardstartperspektive ändern oder zusätzliche Startperspektiven erstellen.

Bei AIR-Mobilprojekten müssen Sie eine Startperspektive vor dem Start der Anwendung konfigurieren.

Weitere Informationen finden Sie unter „Anwendungen ausführen und debuggen“ auf Seite 123.

5 Testen Sie das Projekt.

Testen und optimieren Sie Ihre Projekte mithilfe der Debugging-, Profiling- und Überwachungstools von Flash Builder. In der FlexUnit-Testumgebung können Sie außerdem wiederholbare Tests generieren und bearbeiten, die von Skripten oder direkt von Flash Builder aus ausgeführt werden können.

Weitere Informationen finden Sie unter „Debugging-Tools in Flash Builder“ auf Seite 138 und „Unit-Test-Tools in Flash Builder“ auf Seite 178.

Wenn die Anwendung auf Remotedaten zugreift, überprüfen Sie mithilfe der Netzwerküberwachung die Daten, die zwischen einer Anwendung und einem Datendienst ausgetauscht werden. Weitere Informationen hierzu finden Sie unter „Anwendungen überwachen, die auf Datendienste zugreifen“ auf Seite 230.

Mithilfe des Adobe Flex Profiler können Sie Leistungsengpässe und Speicherlecks in Anwendungen erkennen. Weitere Informationen finden Sie unter „Profiling-Tools in Flash Builder“ auf Seite 149.

6 Stellen Sie eine Anwendung bereit.

Mithilfe von Flash Builder können Sie eine Releaseversion einer Anwendung exportieren, mit der Sie die Anwendung verteilen und bereitstellen können. Der Vorgang zum Exportieren einer Releaseversion einer Anwendung unterscheidet sich je nach Projekttyp.

Weitere Informationen finden Sie unter „Releaseversion einer Anwendung exportieren“ auf Seite 129.

Benutzeroberflächen erstellen

Die Benutzeroberfläche kann zwar vollständig im Quellmodus definiert werden, aber in der Regel fügen Sie zunächst im Designmodus Benutzeroberflächenelemente hinzu.

Weitere Informationen finden Sie unter „Visuell mit Komponenten im MXML-Designmodus arbeiten“ auf Seite 23 und Benutzeroberfläche erstellen.

Hinweis: *Sie können Ihrer Anwendung auch Elemente hinzufügen, die mit Adobe® Flash® Professional erstellt wurden. Siehe „Flash-Komponenten erstellen und bearbeiten“ auf Seite 252.*

Formulare

Mithilfe des Spark-Formulars können Sie Formulare mit einer großen Vielfalt an Möglichkeiten erstellen. Das Spark-Formular ist ein Container, der auf vielfältige Weise angepasst werden kann und mehrere Formulardesigns unterstützt. Sie können außerdem Skins für das Formular und seine Elemente erstellen.

Weitere Informationen zur Verwendung der Container Spark Form, Spark FormHeading und Spark FormItem finden Sie unter Verwenden von Flex 4.6.

Spark-Formular erstellen (Designmodus)

1 Erstellen Sie eine MXML-Komponente. Weitere Informationen finden Sie unter „Benutzerdefinierte MXML-Komponenten erstellen“ auf Seite 239.

2 Fügen Sie im Designmodus des Editors der Komponente ein Spark-Formular hinzu. Wählen Sie im Layoutabschnitt des Komponentenbedienfelds die Formularkomponente aus und ziehen Sie sie in den Designbereich.

3 Legen Sie die Breite und Höhe des Formulars fest. Wählen Sie eine MXML- oder Spark-Skin für die Formularelemente.

Wenn Sie die Skin wählen, wird ein `<fx:Style>`-Block erstellt. Wenn Sie beispielsweise die `StackedFormItemSkin`-Spark-Skin wählen, wird der folgende Stilblock erstellt.

```
<fx:Style>
        @namespace s "library://ns.adobe.com/flex/spark";
        s|Form#form1 s|FormItem
        {
            skinClass: ClassReference("spark.skins.spark.StackedFormItemSkin");
        }
        s|Form#form1 s|FormHeading
        {
            skinClass: ClassReference("spark.skins.spark.StackedFormHeadingSkin");
        }
</fx:Style>
```

Wenn Sie keine Skin angeben, werden die standardmäßigen Skin-Klassen im `spark.skins.spark`-Paket verwendet. Die standardmäßigen Skin-Klassen werden wie folgt verwendet:

- Für das Formular wird die `FormSkin`-Klasse verwendet.
- Für die FormItem-Komponente wird die `FormItemSkin`-Klasse verwendet.
- Für die FormHeading-Komponente wird die `FormHeadingSkin`-Klasse verwendet.

4 Ziehen Sie die FormHeading-Komponente gegebenenfalls in das Spark-Formular. Dabei wird Code wie der folgende generiert:

```
</s:Form>
    <s:FormHeading label="Heading"/>
</s:Form>
```

Doppelklicken Sie im Designmodus auf die FormHeading-Komponente, um die Beschriftung für die Überschrift zu bearbeiten.

5 Ziehen Sie die benötigten Formularkomponenten in das Spark-Formular. Dies könnte z. B. eine TextInput-Komponente sein. Für jede Komponente, die Sie im Formular ablegen, wird ein FormItem erstellt.

Standardmäßig wird jedem FormItem ein `Label`-String hinzugefügt. Wenn Sie z. B. eine TextInput-Komponente in das Formular ziehen, wird Code wie der folgende generiert:

```
<s:Form>
    <s:FormItem label="Label">
        <s:TextInput/>
    </s:FormItem>
</s:Form>
```

Um der TextInput-Komponente einen `helpContent`-String hinzuzufügen, wählen Sie im Eigenschafteninspektor „HelpContent“ aus. Anschließend wird Code wie der folgende generiert:

```
<s:Form>
    <s:FormItem label="Label">
        <s:helpContent>
            <s:Label text="Help String"></s:Label>
        </s:helpContent>
            <s:TextInput/>
    </s:FormItem>
</s:Form>
```

6 In der Eigenschaftenansicht können Sie die Komponenteneigenschaften konfigurieren. Bei Eigenschaften, die auch in der entsprechenden `mx:Form`-Komponente vorhanden sind, sind Funktionsweise und Bedienung ähnlich.

Im Stil-Bedienfeld der Eigenschaftenansicht können Sie die Skins bearbeiten. Weitere Informationen zum Anwenden von Stilen auf Komponenten finden Sie unter „Stile übernehmen“ auf Seite 195.

Sie können außerdem mithilfe einer Vorlage eine benutzerdefinierte MXML-Skin erstellen. Weitere Informationen finden Sie unter „Benutzerdefinierte MXML-Skin für ein Formular anhand einer Vorlage erstellen (Designmodus)“ auf Seite 190.

Skins für Spark-Komponenten generieren und bearbeiten

Die Beispiele unten zeigen, wie Sie benutzerdefinierte MXML-Skins für Spark-Formularelemente erstellen können. Einige Änderungen können Sie im Designmodus des Editors vornehmen, andere erfordern das Bearbeiten der MXML-Datei im Quellmodus. Dabei wird angenommen, dass Sie in einem Flex-Projekt unter Verwendung des standardmäßigen Spark-Designs arbeiten.

Benutzerdefinierte MXML-Skin für Formularelemente anhand einer Vorlage erstellen (Designmodus)

1 Wählen Sie im Designmodus des Editors im Kontextmenü des Formularelements die Option „Skin erstellen“. Das Dialogfeld „Neue MXML-Skin“ wird geöffnet.

   Sie können dieses Dialogfeld auch auf andere Weise öffnen. Wählen Sie im Designmodus des Editors das Formularelement aus. Anschließend haben Sie folgende Möglichkeiten:

   - Wählen Sie in der Eigenschaftenansicht im Abschnitt „Stil“ das Symbol neben dem Feld „Skin“ und anschließend „Skin erstellen“.
   - Wählen Sie im Menü „Datei“ die Befehle „Neu“ > „MXML-Skin“.
   - Wählen Sie im Menü „Design“ den Befehl „Skin erstellen“.

2 Geben Sie im Dialogfeld „Neue MXML-Skin“ Folgendes an.

   - Quellordner und Paket für die generierte Skin.
   - Den Namen für die Skinklasse, die Sie erstellen.
   - Die Hostkomponente wird standardmäßig ausgewählt.
   - (Empfohlen) Aktivieren Sie das Kontrollkästchen „Erstellen als Kopie von“ und deaktivieren Sie nicht das Kontrollkästchen „ActionScript-Stilcode entfernen“. Wenn Sie mit Skinning nicht vertraut sind, verwenden Sie zum Erstellen einer Skinklasse eine Kopie.

     (Fortgeschrittene Benutzer) Wenn Sie mit den Schritten zum Erstellen von Skinklassen vertraut sind, gehen Sie folgendermaßen vor:

     Aktivieren Sie die Option zum Entfernen von ActionScript-Stilcode oder deaktivieren Sie die Option zum Erstellen einer Kopie einer vorhandenen Klasse. Wenn Sie keine Kopie einer vorhandenen Klasse erstellen, generiert Flash Builder eine leere Skinklassendatei mit einigen nützlichen Hinweisen.

     Die verbleibenden Schritte dieses Vorgangs setzen voraus, dass Sie die empfohlene Option zum Generieren einer Skinklasse gewählt haben. Weitere Informationen finden Sie unter „Skins für Spark-Komponenten generieren und bearbeiten“ auf Seite 208.

3 Klicken Sie auf „Fertig stellen“. Flash Builder generiert eine Skinklassendatei und öffnet sie im Designmodus des MXML-Editors. Sie können die Skin mithilfe der Bearbeitungswerkzeuge im Bedienfeld „Stil“ der Eigenschaftenansicht bearbeiten.

4 Sie werden aufgefordert, die neue Skin auf alle Formularelemente anzuwenden. Wenn Sie dies tun, wird ein `<fx:Style>`-Block erstellt, der die neue Skin angibt.

   Wenn Sie z. B. die `SkinnableContainerSkin`-Skin auf alle FormItems anwenden, wird Code wie der folgende erstellt:

```
<fx:Style>
    @namespace s "library://ns.adobe.com/flex/spark";
    s|Form#form1 s|FormItem
    {
        skinClass: ClassReference("spark.skins.spark.SkinnableContainerSkin");
    }
</fx:Style>
<s:Form id="form1" x="145" y="81">
    <s:FormHeading label="Heading"/>
    <s:FormItem label="Label">
        <s:helpContent>
            <s:Label text="Help String"></s:Label>
        </s:helpContent>
        <s:TextInput/>
    </s:FormItem>
    <s:FormItem label="Label">
        <s:helpContent>
            <s:Label text="Help String"></s:Label>
        </s:helpContent>
        <s:TextInput/>
    </s:FormItem>
</s:Form>
```

Wenn Sie die Skin jedoch nur auf das ausgewählte Formularelement anwenden, müssen alle Formularelemente mit dieser Skin dieselbe Anzahl Spalten enthalten.

Wenn Sie z. B. die `SkinnableContainerSkin`-Skin auf nur ein FormItem anwenden, wird Code wie der folgende erstellt:

```
<s:Form x="186" y="127">
    <s:FormHeading label="Heading"/>
    <s:FormItem label="Label" skinClass="spark.skins.spark.SkinnableContainerSkin">
        <s:helpContent>
            <s:Label text="Help String"></s:Label>
        </s:helpContent>
        <s:TextInput/>
    </s:FormItem>
    <s:FormItem label="Label">
        <s:helpContent>
            <s:Label text="Help String"></s:Label>
        </s:helpContent>
        <s:TextInput/>
    </s:FormItem>
</s:Form>
```

Mehrere Skins für verschiedene Formularelemente im selben Formular definieren (Designmodus)

Berücksichtigen Sie die folgenden Punkte, wenn Sie mehrere Skinklassen für Formularelemente in einem Formular definieren.

- Es wird empfohlen, bei Formularen mit mehreren Formularelementen dieselbe Skinklasse für alle Formularelemente zu verwenden. Sie brauchen dieser Empfehlung jedoch nicht unbedingt zu folgen.
- Die Formularelemente innerhalb eines Formulars müssen dieselbe Anzahl Spalten enthalten.

Gehen Sie wie folgt vor, um eine Skinklasse zu definieren:

1 Wählen Sie das Formularelement im Designmodus des Editors aus.

2 Klicken Sie im Abschnitt „Stil“ der Eigenschaftenansicht auf „Skin“ und wählen Sie eine Skinklasse. Eine Warnmeldung bezüglich der empfohlenen Verwendung wird angezeigt. Wählen Sie „Nur auf ausgewählte Elemente anwenden“.

3 Führen Sie diese Schritte für alle Formelemente aus, auf die Sie die Skin anwenden möchten.

Skins für Spark-Formulare generieren und bearbeiten

Die Beispiele unten zeigen, wie Sie benutzerdefinierte MXML-Skins für Spark-Formulare erstellen können. Dabei wird angenommen, dass Sie in einem Flex-Projekt unter Verwendung des standardmäßigen Spark-Designs arbeiten.

Benutzerdefinierte MXML-Skin für ein Formular anhand einer Vorlage erstellen (Designmodus)

Um eine benutzerdefinierte MXML-Skin für das Formular zu erstellen, öffnen Sie das Dialogfeld „Neue MXML-Skin“. Dazu haben Sie folgende Möglichkeiten:

Wählen Sie das Formular im Designmodus des Editors aus und führen Sie einen der folgenden Vorgänge aus:

- Wählen Sie im Kontextmenü des Formulars die Option „Skin erstellen“.
- Wählen Sie in der Eigenschaftenansicht im Abschnitt „Stil“ das Symbol neben dem Feld „Skin“ und anschließend „Skin erstellen“.
- Wählen Sie im Menü „Datei“ die Befehle „Neu“ > „MXML-Skin“.
- Wählen Sie im Menü „Design“ den Befehl „Skin erstellen“.

Die übrigen Schritte dieses Vorgangs sind denen beim Erstellen von Skins für Formularelemente ähnlich. Weitere Informationen finden Sie unter „Benutzerdefinierte MXML-Skin für Formularelemente anhand einer Vorlage erstellen (Designmodus)“ auf Seite 188.

Eine einheitliche Skin für alle Formularelemente eines Formulars definieren (Designmodus)

Um eine einzelne Skinklasse für sämtliche FormItems in einem Formular zu definieren, definieren Sie die FormItem-Skin im Formular. Gehen Sie dazu wie folgt vor:

1 Wählen Sie im Designmodus des Editors das Formular aus.

2 Wählen Sie im Abschnitt „Allgemein“ der Eigenschaftenansicht eine FormItem-Skin.

Die ausgewählte Skin wird auf alle FormItems angewendet und ein `<fx:Style>`-Block wird erstellt. Wenn Sie z. B. die StackedFormItemSkin für die FormItems in einem Formular wählen, wird der `<fx:Style>`-Block wie folgt generiert:

```
<fx:Style>
@namespace s "library://ns.adobe.com/flex/spark";
s|Form#form1 s|FormItem
{
    skinClass: ClassReference("spark.skins.spark.StackedFormItemSkin");
}
s|Form#form1 s|FormHeading
{
    skinClass: ClassReference("spark.skins.spark.StackedFormHeadingSkin");
}
</fx:Style>
```

Diagramme

Mit Flash Builder können Sie Diagrammkomponenten hinzufügen, um Daten in der Benutzeroberfläche anzuzeigen. Mit den Flex-Diagrammkomponenten können Sie einige der am häufigsten verwendeten Diagrammtypen erstellen und das Layout der Diagramme flexibel anpassen. Eine Übersicht der verschiedenen verfügbaren Diagramme finden Sie unter Chart types.

Informationen zum Definieren von Diagrammdaten, Formatieren von Diagrammelementen sowie zum Ändern anderer Diagrammaspekte finden Sie unter Einführung in Diagramme.

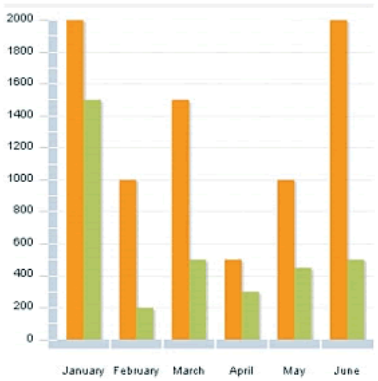
Diagrammkomponente hinzufügen

1 Ziehen Sie im Designmodus des MXML-Editors eine Diagrammkomponente aus der Komponentenansicht in den Designbereich.

2 Geben Sie eine ID für das Diagramm ein.

3 Um mehr als eine Datenreihe im Diagramm anzuzeigen, klicken Sie auf die Schaltfläche „Hinzufügen“ und geben Sie im angezeigten Dialogfeld den neuen Reihennamen ein.

Das Steuerelement „ColumnChart“ hat beispielsweise zwei Datenreihen. Der Balken auf der linken Seite gibt den Bruttogewinn für 6 Monate an und der Balken daneben den Nettogewinn für den gleichen Zeitraum.



Entfernen Sie eine Datenreihe, indem Sie sie in der Liste auswählen und auf die Schaltfläche „Entfernen“ klicken.

4 (Optional) Aktivieren Sie die Option „Legende einschließen“.

Wenn Sie die Option „Legende einschließen“ aktivieren, können Sie ein Legenden-Steuerelement zum Diagramm hinzufügen, das die Beschriftung für jede Datenreihe im Diagramm sowie eine Legende mit dem Diagrammelement für die Reihe anzeigt.

5 Klicken Sie auf „OK“, um das Diagramm einzufügen.

Designs übernehmen

Mit Designs können Sie Anwendungen persönlicher gestalten. Flash Builder enthält mehrere Designs, aus denen Sie wählen können. Sie können zusätzliche Designs importieren oder eigene Designs erstellen.

Flash Builder enthält einen Satz von Spark-Designs und einen Satz von Halo-Designs. Das standardmäßige Design für Flex 4.6- und Flex 4-Komponenten ist Spark. Halo ist das standardmäßige Design für Flex 3.

Weitere Informationen zur Unterstützung von Designs in Flex finden Sie unter About themes.

Designs auswählen

Legen Sie Designs auf Projektbasis fest Nachdem Sie ein Design für ein Projekt festgelegt haben, verwenden alle Anwendungen im Projekt das gleiche Design.

1 Öffen Sie das Dialogfeld „Projektdesign auswählen“ (entweder von der Designansicht oder der Quellansicht des MXML-Editor aus):

- (Designansicht) Wählen Sie das Aussehen-Bedienfeld aus. Wählen Sie dann das aktuelle Design aus.
- (Quellansicht) Wählen Sie im Flash Builder-Menü „Projekt“ die Option „Eigenschaften“ -> „Flex Design“ aus.

2 Wählen Sie ein Design aus und klicken Sie dann auf „OK“.

Mobildesigns

Wenn Sie in Flex ein mobiles Projekt erstellen, wird das standardmäßige Mobildesign verwendet. Wie Sie das standardmäßige Mobildesign ändern können, erfahren Sie unter Mobildesigns.

Für Mobildesigns stehen weniger Stile zur Auswahl als für andere Designs. Eine vollständige Liste der Stile für Mobildesigns finden Sie unterStile für Mobildesigns.

Designs importieren

Sie können Flash Builder zum Importieren von Designs verwenden. Die Dateien für ein Design müssen sich in einem Ordner befinden. Alle für ein Flex Design erforderlichen Dateien müssen vorhanden sein.

Der Name des Designs ist durch das Namenselement in der Datei `metadata.xml` festgelegt, die sich im Designordner befindet. Wenn das Namenselement nicht festgelegt ist oder wenn `metadata.xml` nicht vorhanden ist, wird als Designname der Name des Designordners verwendet.

Weitere Informationen zum erforderlichen Format für Flex Designs finden Sie unter „Designs erstellen“ auf Seite 193.

Flash Builder-Designs können in den folgenden Formaten vorliegen:

- Komprimierte Designdatei (ZIP)

  Extrahieren Sie den Inhalt der ZIP-Datei, bevor Sie das Design importieren. Der extrahierte Inhalt muss alle erforderlichen Dateien enthalten.

- CSS- oder SWC-Datei für ein Design

  Die CSS- oder SWC-Datei muss in einem Ordner sein, der alle für ein Flex Design erforderlichen Dateien enthält. Wenn Sie über Flash Builder ein Design importieren, wählen Sie entweder die CSS- oder die SWC-Datei für das Design aus.

- MXP-Datei

  Sie können Adobe Extension Manager CS5 verwenden, um Dateien für Flex Designs in einer MXP-Datei zu verpacken. Das Design kann dann über den Extension Manager in Flash Builder importiert werden.

  Weitere Informationen zum Verpacken von Designs in einer MXP-Datei finden Sie unter „Erweiterungsdateien (MXP-Datei) für Flex Design erstellen“ auf Seite 194.

Flex Designs über Flash Builder importieren

1 Öffen Sie das Dialogfeld „Projektdesign auswählen“ (entweder von der Designansicht oder der Quellansicht des MXML-Editor aus):

- (Designansicht) Wählen Sie das Aussehen-Bedienfeld aus. Wählen Sie dann das aktuelle Design aus.
- (Quellansicht) Wählen Sie im Flash Builder-Menü „Projekt“ die Option „Eigenschaften“ -> „Flex Design“ aus.

2 Klicken Sie auf „Design importieren“, wählen Sie den Ordner mit dem Design aus, das importiert werden soll, wählen Sie die CSS- oder SWC-Datei aus und klicken Sie auf „OK“.

In MXP-Dateien verpackte Flex Designs importieren

1 Importieren Sie Adobe Flash® Builder™ 4.6 in Adobe Extension Manager CS5 (sofern Sie dies nicht bereits getan haben):

Wählen Sie im Adobe Extension Manager die Option „Datei“ > „Produkt importieren“ aus.

2 Öffnen Sie Adobe Extension Manager und wählen Sie „Flash Builder 4.6“ aus.

3 Wählen Sie „Datei“ > „Erweiterung installieren“, navigieren Sie zur MXP-Datei für das Design und klicken Sie auf „Öffnen“.

Nachdem Sie die Lizenz akzeptiert haben, installiert Adobe Extension Manager das Design in Flash Builder. Das Design ist jetzt in Flash Builder über das Dialogfeld „Projektdesign auswählen“ verfügbar.

Hinweis: *Sie können auch auf die MXP-Datei doppelklicken, um Adobe Extension Manager zu starten, der das Design dann automatisch installiert.*

Designs herunterladen

Sie können Designs herunterladen, die dann in Flash Builder importiert werden können.

Flex Designs herunterladen

1 Öffen Sie das Dialogfeld „Projektdesign auswählen“ (entweder von der Designansicht oder der Quellansicht des MXML-Editor aus):

- (Designansicht) Wählen Sie das Aussehen-Bedienfeld aus. Wählen Sie dann das aktuelle Design aus.
- (Quellansicht) Wählen Sie im Flash Builder-Menü „Projekt“ die Option „Eigenschaften“ -> „Flex Design“ aus.

2 Klicken Sie auf „Mehr Designs finden“.

Flash Builder öffnet den Standardwebbrowser mit einer Seite, die Designs zum Download enthält. Sie können auch zu anderen Sites navigieren, die herunterladbare Flex Designs enthalten.

3 Wählen Sie ein Flex Design aus, das heruntergeladen werden soll.

Sobald Sie das Design heruntergeladen haben, können Sie es wie unter „Designs importieren“ auf Seite 192 beschrieben importieren.

Designs erstellen

Sie können eigene Designs erstellen und sie dann in Flash Builder importieren. Ein Flex Design enthält typischerweise folgende Dateien:

- SWC, SWF-Datei, CSS-, PNG-, JPEG- und andere Dateien, die zusammen das Design bilden.

  Aus welchen Dateien das Design besteht, kann variieren, aber es muss eine SWC- oder CSS-Datei vorhanden sein.

- `Datei „preview.jpg"`

  Die Vorschaubilddatei für das Design. Wenn der Designordner keine Datei mit dem Namen `preview.jpg` enthält, verwendet Flash Builder ein Standardvorschaubild für das Design.

- `Datei „metadata.xml"`

  Enthält Informationen zum Design wie z. B. SDK-Versionen, mit denen das Design kompatibel ist. Wenn der Designordner diese Datei nicht enthält, erstellt Flash Builder beim Importieren des Designs eine Datei.

Typischerweise verpacken Sie ein Design in einer ZIP-Datei; die ZIP-Datei muss aber extrahiert werden, bevor Sie das Design in Flash Builder importieren können. Sie können die Designdateien auch in einer MXP-Datei (Adobe Extension Manager) verpacken und Adobe Extension Manager verwenden, um das Design in Flash Builder zu importieren.

Weitere Informationen finden Sie unter About themes.

Datei „metadata.xml“

In der folgenden Tabelle sind die Elemente aufgeführt, die in die Datei `metadata.xml` aufgenommen werden können.

Elementname	Beschreibung
Name	Der Name des Designs, das in Flash Builder angezeigt wird. Wenn Sie über Flash Builder ein Design importieren, können Sie den in der Datei `metadata.xml` angegebenen Namen überschreiben.
category	Autor des Designs. Die Kategorie, unter der das Design in Flash Builder angezeigt wird.
sdks	Gibt die Flex SDK-Versionen an, mit denen das Design kompatibel ist. Dieses Element ist ein übergeordnetes Element für „minVersionInclusive“ und „maxVersionExclusive“. Wenn das Element „sdks“ fehlt, ist das Design für alle SDKs gültig.
minVersionInclusive	Älteste Flex SDK-Version, mit der dieses Design kompatibel ist. Wenn dieses Element fehlt, ist das Design mit allen älteren Versionen des Flex SDKs kompatibel.
maxVersionExclusive	Neueste SDK-Version, mit der dieses Design kompatibel ist. Wenn dieses Element fehlt, ist das Design mit allen neueren Versionen des Flex SDKs kompatibel.
mainFile	Oberste Datei für die Implementierung des Designs. Diese Datei kann andere Dateien im Design referenzieren. Beispielsweise könnte eine CSS-Datei eine SWC- oder SWF-Datei referenzieren. Das Compiler-Argument „-theme“ referenziert die angegebene Datei.

Das folgende Beispiel zeigt eine typische `metadata.xml`-Datei für ein Design, das von Firma ABC erstellt wurde.

```
<theme>
    <name>WindowsLookAlike</name>
    <category>ABC</category>
        <sdks>
            <minVersionInclusive>2.0.1</minVersionInclusive>
            <maxVersionExclusive>4.0.0</maxVersionExclusive>
        </sdks>
    <mainFile>WindowsLookAlike.css</mainFile>
</theme>
```

Laut der Datei `metadata.xml` ist das Design mit dem Flex 2.0.1 SDK kompatibel. Es ist auch mit älteren SDKs als Flex 4.0.0 kompatibel. Wenn dieses Design ausgewählt wird, wird die Datei `WindowsLookAlike.css` dem Compiler-Argument `-themes` hinzugefügt.

Erweiterungsdateien (MXP-Datei) für Flex Design erstellen

Sie können Adobe Extension Manager CS5 verwenden, um eine MXP-Datei (Extension) für ein Flex Design zu erstellen. Die MXP-Datei kann über Adobe Extension Manager CS5 in Flash Builder importiert werden.

Platzieren Sie alle Designdateien in einem Staging-Ordner und erstellen Sie dann eine Erweiterungsinstallationsdatei (MXI-Datei), die vom Extension Manager zum Erstellen der MXP-Datei verwendet wird. Weitere Informationen zum Format einer MXI-Datei finden Sie im Dokument Erweiterungsdateiformat.

Wenn Sie die MXI-Datei erstellen, legen Sie Zielpfade für alle Designdateien fest. Die Zielpfade sind im folgenden Format:

```
$flexbuilder/<Theme Name>
```

- `$flexbuilder` ist in der Flash Builder-Konfigurationsdatei `XManConfig.xml` definiert. Extension Manager erweitert `$flexbuilder` entsprechend dieser Definition. `XManConfig.xml` befindet sich im folgenden Pfad im Dateisystem:

  ```
  /<Install Dir>/Flash Builder 4/configuration/XManConfig.xml
  ```

- *<Designname>* ist der Name des Ordners, der das Flex Design enthält.

MXP Extension-Dateien für ein Flex Design erstellen

1. Platzieren Sie alle Dateien für das Design einschließlich der MXI-Datei in einem Staging-Ordner.
2. Wählen Sie im Extension Manager die Option „Datei“ > „Erweiterung erstellen“ aus.
3. Wechseln Sie zu dem Ordner, der die Extension-Installationsdatei enthält, und wählen Sie sie aus.
4. Wechseln Sie zu einem Ordner für die Paketdatei und geben Sie einen Namen sowie die Erweiterung .mxp ein.

   Sie können dann die Erweiterungsdatei testen, indem Sie sie über den Extension Manager installieren.

Weitere Designs hinzufügen

Sie können mehr als eine Designdatei angeben, die auf eine Anwendung angewendet wird. Wenn keine überlappenden Stile vorhanden sind, werden beide Designs vollständig angewendet. Beim Hinzufügen zusätzlicher Designs sind weitere Aspekte zu berücksichtigen, z. B. das Ordnen der Designdateien.

Um zusätzliche Designs hinzuzufügen, verwenden Sie den Befehlszeilen-Compiler `mxmlc` mit der `theme`-Compiler-Option, um den Pfad zu den Designdateien anzugeben.

In Using themes finden Sie Details zum Angeben von Compiler-Argumenten und Ordnen von Designdateien.

Stile übernehmen

Stile wirken sich auf das Aussehen einer Anwendung aus, indem sie Werte für die visuellen Parameter von Anwendungskomponenten vorgeben. Sie können Stile einstellen, die für alle Komponenten einer Anwendung, einzelne Komponenten oder einen Satz von Komponenten gelten, die von einer Stilauswahl referenziert werden.

Beispielsweise können Sie Stile wie die folgenden angeben:

- Text

  Schriftfamilie, Größe, Stärke, Farbe, Schriftart (fett, kursiv, unterstrichen) und andere Textanzeigeeinstellungen

- Rahmen

  Stärke, Farbe, Rollover-Farbe, Rahmenstil (durchgehend, nach innen versetzt, nach außen versetzt, ohne), Eckenradius und andere

- Farbe

  Füllfarbe und Alpha

Hinweis: *Die für eine Komponente verfügbaren Stile variieren je nach Komponente.*

Sie legen Stileigenschaften in einem MXML-Tag oder separat über CSS-Code fest. Der CSS-Code kann sich zwischen `<fx:Style>`-Tags in einer Anwendung oder in einer separaten CSS-Datei befinden.

Wenn Sie eingebundene Stile auf Komponenten anwenden, können Sie Komponentenstile in eine CSS-Regel in einem externen Stylesheet konvertieren. Mit dem CSS-Editor können Sie CSS-Dateien bearbeiten.

Sie können auch Skins, die auf eine Komponente angewendet werden, in Stile konvertieren.

Verwenden Sie den Designmodus des MXML-Editors, um Stile auf eine Anwendung oder spezifische Anwendungskomponenten anzuwenden. Im Designmodus können Sie außerdem angewendete Stile oder Skins in CSS-Stylesheets konvertieren.

Vergleich von Stil und Skin

Skinning ist der Vorgang, mit dem das Aussehen einer Komponente durch Modifizieren oder Ersetzen ihrer visuellen Elemente geändert wird. Diese Elemente können aus Bitmapbildern, SWF-Dateien oder Klassendateien bestehen, die Methoden zur Definition von Vektorbildern enthalten. Skins können das Aussehen einer Komponente in verschiedenen Status definieren. Beispielsweise können Sie das Aussehen einer Button-Komponente in den Status „Normal“, „Gedrückt“, „Darüber“ und „Deaktiviert“ angeben.

Mit Flash Builder können Sie die Skin einer Komponente in CSS-Stile konvertieren.

Stile auf eine Anwendung anwenden

Verwenden Sie die Ansicht „Aussehen“, um Stile zu definieren, die für eine ganze Anwendung gelten. Flash Builder speichert den in der Ansicht „Aussehen“ definierten Stil als globale CSS-Stilauswahl.

1 Öffnen Sie im Designmodus des MXML-Editors eine MXML-Anwendungsdatei, die mehrere Komponenten enthält.

2 Geben Sie in der Ansicht „Aussehen“ globale Stile für die Anwendung an.

   Wenn die Ansicht „Aussehen“ nicht angezeigt wird, wählen Sie „Fenster“ > „Ansicht anzeigen“ >„Andere“ > „Flash Builder“ > „Aussehen“.

   Wenn Sie die Stile angewendet haben, erstellt Flash Builder eine globale CSS-Stilauswahl für den angegebenen Stil.

   Wenn dies der erste Stil ist, auf den in der Anwendung verwiesen wird, erstellt Flash Builder eine neue CSS-Datei und verweist in der MXML-Anwendungsdatei darauf.

   Wenn Ihre Anwendung bereits auf eine CSS-Datei oder einen `<fx:Style>`-Block verweist, aktualisiert Flash Builder die CSS-Datei mit der globalen Stilauswahl.

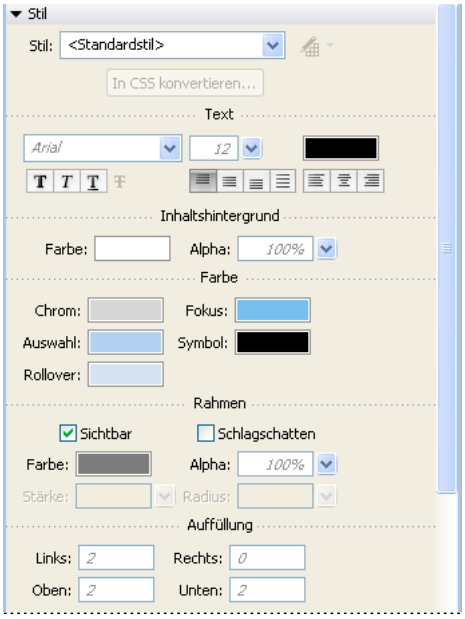
Eingebundene Stile auf eine Komponente anwenden

Verwenden Sie die Eigenschaftenansicht, um eingebundene Stile für ausgewählte Komponenten zu definieren.

1 Öffnen Sie im Designmodus des MXML-Editors eine MXML-Anwendungsdatei, die mehrere Komponenten enthält.

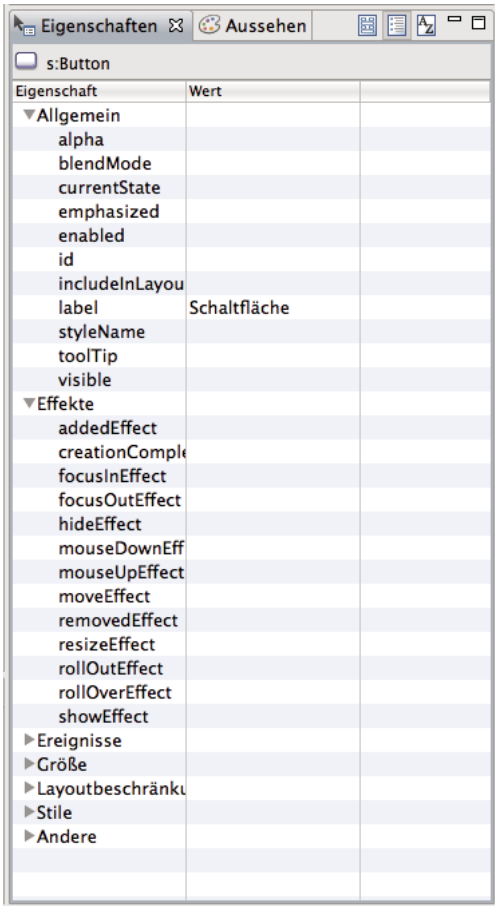
2 Wählen Sie eine Komponente aus und geben Sie im Bereich „Stil“ der Eigenschaftenansicht Werte für die Stileigenschaften an.

   Je nach der gewählten Komponente ändert sich der Bereich „Stil“.



Eigenschaftenansicht, in der Stile für ein DataGrid angezeigt werden

3 Wählen Sie in der Eigenschaftenansicht die Kategorieansicht, um alle Stile aufzulisten, die auf die gewählte Komponente angewendet werden können.



Hinweis: *Aus mehreren Wörtern zusammengesetzte Stilnamen in Flex können entweder wie eine ActionScript-ID (z. B. fontFamily) oder wie ähnliche HTML-Stile (z. B. font-family) geschrieben werden.*

4 Wechseln Sie nach dem Angeben eines Stils in den Quellmodus, um den generierten Code anzuzeigen, der für den Stil gilt.

Einen externen oder eingebetteten Stil auf eine Anwendung anwenden

Sie können CSS-Stile in eine MXML-Anwendungsdatei einbetten oder auf eine externe CSS-Datei verweisen. Das folgende Beispiel zeigt CSS-Code, durch den Stile auf alle Spark-Button-Komponenten einer Anwendung angewendet werden. Es wird auch der Selektor`myStyle` erstellt, der auf jede Komponente angewendet werden kann:

```
@namespace s "library://ns.adobe.com/flex/spark";
@namespace mx "library://ns.adobe.com/flex/mx";

s|Button { fontSize: 16pt; color: Red } /* type selector */
.myStyle { color: Red } /* class selector */
```

Für Stile, die auf Komponenten angewendet werden, wie z. B. `s|Button`, muss der Selektor für Komponenten einen Namespace angeben. In diesem Beispiel definiert s|Button einen Stil, der automatisch auf alle Spark-Button-Komponenten angewendet wird.

Verwenden Sie die CSS-Punktschreibweise, um einen Selektor zu erstellen, der auf beliebige Komponenten angewendet werden kann. In diesem Beispiel braucht `.myStyle` keinen Namespace zu deklarieren und kann auf beliebige Komponenten angewendet werden.

Flex hat spezifische Anforderungen für das Erstellen und Anwenden von Stilen. Weitere Informationen finden Sie unter Using styles in Flex.

Stile auf eine Anwendung anwenden

1 Öffnen Sie im Designmodus des MXML-Editors eine MXML-Anwendungsdatei, die mehrere Spark-Button-Komponenten und eine CheckBox-Komponente enthält.

2 (Eingebettete Stile) Fügen Sie im Quellmodus den folgenden Code zu Ihrer Anwendung hinzu:

```
<fx:Style>
    @namespace s "library://ns.adobe.com/flex/spark";
    @namespace mx "library://ns.adobe.com/flex/halo";

    s|Button { fontSize: 16pt; color: Red } /* type selector */
    .myStyle { fontSize: 16pt; color: Blue } /* class selector */
</fx:Style>
```

3 (Externes Stylesheet) Erstellen Sie ein Stylesheet, das die `s|Button`- und `.myStyle`-Selektoren in Schritt 1 implementiert. Verweisen Sie im Quellmodus auf die Datei aus der MXML-Anwendungsdatei:

```
<fx:Style source="styles.css"/>
```

4 Wechseln Sie in den Designmodus. Der Stil ist bereits auf die Spark-Buttons angewendet.

5 Aktivieren Sie das Kontrollkästchen „Komponente“ und wählen Sie anschließend in der Ansicht „Eigenschaften“ die Option „Stile“ > „myStyle“.

Auf die CheckBox-Komponente wurde jetzt der`myStyle`-Stil angewendet.

In CSS konvertieren

Sie können eingebundene Stile und Komponentenskins in CSS-Stile konvertieren. Bei der Konvertierung können Sie angeben, ob Stile global oder für eine bestimmte Komponente gelten sollen. Außerdem können Sie einen CSS-Stilselektor für die generierten Stile angeben.

1 Wählen Sie im Designmodus des MXML-Editors eine Komponente im Designbereich aus. Verwenden Sie Inline-Stile, oder geben Sie eine skinClass-Eigenschaft für die Komponente an.

Informationen zum Angeben einer skinClass für eine Komponente finden Sie unter „Skins für Spark-Komponenten generieren und bearbeiten“ auf Seite 208.

2 Klicken Sie in der Eigenschaftenansicht auf „In CSS konvertieren“.

3 Wenn Sie mehrere Projekte im Arbeitsbereich geöffnet haben, wählen Sie nur die Ressourcendateien aus, die Sie im Dialogfeld „Ressource speichern“ speichern möchten. Klicken Sie dann auf „OK“.

4 Wählen Sie im Dialogfeld „Neue Stilregel“ die CSS-Datei aus oder klicken Sie auf „Neu“, um eine neue Datei zu erstellen.

5 Geben Sie den Selektortyp an. Wählen Sie aus Folgendem:

- Alle Komponenten

  Der Stil ist global und wird auf alle Komponenten in der Anwendung angewendet.

- Alle Komponenten mit Stilnamen

Komponenten geben diese Stilauswahl mithilfe des styleName-Attributs an. Wenn Sie diese Option wählen, geben Sie einen Namen für den Selektor an.

- Bestimmte Komponente

  Der Stil gilt nur für die gewählte Komponente.

- Bestimmte Komponente mit Stilnamen

  Der Stil gilt nur für die gewählte Komponente und verweist über den Namen des Typselektors auf den Stil. Wenn Sie diese Option wählen, geben Sie einen Namen für den Typselektor an.

6 Klicken Sie auf „OK“.

Flash Builder generiert oder aktualisiert die angegebene CSS-Datei. Flash Builder ändert auch den Quellcode in der Anwendung, um auf den Typselektor in der CSS-Datei zu verweisen.

Flash Builder entfernt Verweise auf den eingebundenen Stil oder die `skinClass`-Eigenschaft der Komponente.

Stilregel bearbeiten

Wenn bereits externe CSS-Stile auf eine Komponente angewendet sind, können Sie schnell von der Komponente aus die betreffenden Stile bearbeiten.

1 Wählen Sie eine Komponente aus.

2 Klicken Sie in der Eigenschaftenansicht neben dem Popupmenü „Stil“ auf die Schaltfläche „Stilregel bearbeiten“ und wählen Sie dann den Stil aus, den Sie bearbeiten möchten.

Die CSS-Datei wird im Designmodus des CSS-Editors geöffnet. Verwenden Sie die Eigenschaftenansicht, um weitere Änderungen vorzunehmen. Sie können das CSS auch im Quellmodus ändern.

CSS-Dateien erstellen

Mit dem Assistenten zum Erstellen neuer CSS-Dateien erstellen Sie CSS-Dateien für ein Projekt. Der Assistent zum Erstellen neuer CSS-Dateien erstellt eine leere Datei, mit der Sie Ihre CSS-Stile definieren können.

Standardmäßig fügt Flash Builder die Standard-Namespaces für Spark- und MX-Stile hinzu.

So erstellen Sie eine neue leere CSS-Datei:

1 Wählen Sie im Flash Builder-Menü „Datei“ > „Neue CSS-Datei“.

2 Geben Sie einen Quellordner an.

Der Quellordner kann sich im aktuellen Projekt oder in einem anderen Projekt befinden.

3 Geben Sie ein Paket für die Datei an.

Wählen Sie aus den im Projekt verfügbaren Paketen. Wenn Sie die Datei in einem neuen Paket ablegen möchten, müssen Sie zunächst ein Paket erstellen. Wählen Sie „Datei“ > „Neues „Paket“.

4 Geben Sie einen Namen für die Datei an.

5 Klicken Sie auf „Fertig stellen“.

Flash Builder verwendet Vorlagen, die den Inhalt von neu erstellten Dateien definieren. Sie können die von Flash Builder verwendeten Vorlagen anpassen. Siehe „Dateivorlagen anpassen“ auf Seite 63.

CSS-Editor

Flash Builder bietet einen CSS-Editor, mit dem Sie Stylesheets für Ihre Anwendung erstellen und bearbeiten können. Der CSS-Editor ist ausschließlich im Quellmodus verfügbar.

Wenn Sie in Flash Builder ein Stylesheet erstellen, stellt Flash Builder die folgenden Deklarationen für die Spark- und MX-Namespaces bereit:

```
@namespace s "library://ns.adobe.com/flex/spark";
@namespace mx "library://ns.adobe.com/flex/mx";
```

Einige Spark- und MX-Komponenten haben denselben lokalen Namen. Beispielsweise gibt es eine Spark-Button-Komponente (im spark.components.*-Paket) und eine MX-Button-Komponente (im mx.controls.*-Paket). Um verschiedene Komponenten mit demselben Namen voneinander zu unterscheiden, geben Sie in Ihrem CSS Namespaces an, die auf Typen angewendet werden.

Wenn Sie in Ihren Stylesheets keine Typselektoren verwenden, müssen Sie keine Namespaces deklarieren. Weitere Informationen und Beispiele finden Sie unter About namespaces in CSS.

Hinweis: *Bei Anwendungen, die das Flex 3 SDK verwenden, werden Stile anders verarbeitet. Wenn Sie in einem Flex-Projekt arbeiten, das das Flex 3 SDK angibt, nimmt der CSS-Editor das Verhalten an, das in Flex Builder 3 implementiert ist. Informationen zur Verwendung des CSS-Editors für Anwendungen, die das Flex 3-SDK verwenden, finden Sie in der Dokumentation zu Flex Builder 3.*

Anzeigestatus und Übergänge hinzufügen

Mit Adobe® Flash® Builder™ können Sie Anwendungen erstellen, deren Benutzeroberfläche sich je nach den vom Benutzer ausgeführten Arbeitsschritten ändert. Beispielsweise könnte der Basisstatus der Anwendung die Homepage sein und ein Logo, eine Seitenleiste und Begrüßungstext enthalten. Wenn der Benutzer auf eine Schaltfläche in der Seitenleiste klickt, ändert sich die Benutzeroberfläche (d. h. ihr *Status*) dynamisch und im Hauptbereich wird jetzt ein Bestellformular angezeigt. Das Logo und die Seitenleiste sind weiterhin zu sehen.

In Flex können Sie diese Interaktion mit Anzeigestatus und Übergängen hinzufügen. Ein *Ansichtsstatus* ist eine von mehreren Ansichten, die Sie für eine Anwendung oder eine benutzerdefinierte Komponente definieren. Ein *Übergang* besteht aus einem oder mehreren Effekten, die zusammen abgespielt werden, wenn sich ein Ansichtsstatus ändert. Zweck eines Übergangs ist, für einen visuell glatten Wechsel von einem Status zum nächsten zu sorgen.

Ansichtsstatus und Übergänge

Ein *Ansichtsstatus* ist eines von mehreren Layouts, das Sie für eine einzelne MXML-Anwendung oder Komponente definieren. Sie erstellen eine Anwendung oder Komponente, die je nach den Aktionen des Benutzers von einem Ansichtsstatus zu einem anderen wechselt. Sie können Ansichtsstatus verwenden, um eine Benutzeroberfläche zu erstellen, die der Benutzer anpassen kann, oder die weitere Informationen anzeigt, wenn der Benutzer bestimmte Arbeitsschritte abgeschlossen hat.

Jede in einer MXML-Datei definierte Anwendung oder Komponente hat mindestens einen Status (den *Basisstatus*), der im Standardlayout der Datei angezeigt wird. Sie können einen Anfangsstatus als Ablagebereich für Inhalt wie z. B. Navigationsleisten oder Logos verwenden, der in allen Ansichten in einer Anwendung oder Komponente enthalten ist, um für ein konsistentes Layout zu sorgen.

Sie erstellen einen Asichtsstatus, indem Sie das Layout eines vorhandenen Status ändern oder ein neues Layout erstellen. Änderungen an einem vorhandenen Status können aus der Bearbeitung, Verschiebung, Hinzufügung oder Löschung von Komponenten bestehen. Das neue Layout ist das Layout, das Benutzer beim Wechsel zu einem anderen Status sehen.

Einen vollständigen Überblick über das Konzept der Ansichtsstatus sowie Beispiele finden Sie unter View states.

Anders als bei einer HTML-basierten Anwendung werden in einer mit Flex erstellten Anwendung keine Seiten hinzugefügt. Sie erstellen eine einzige MXML-Anwendungsdatei und fügen dann verschiedene Layouts hinzu, zu denen beim Ausführen der Anwendung gewechselt werden kann. Sie können für diese Layouts Ansichtsstatus, aber auch den ViewStack-Navigatorcontainer mit anderen Navigatorcontainern verwenden.

Wenn Sie die Ansichtsstatus in der Anwendung ändern, ändert sich auch die Benutzeroberfläche. Standardmäßig scheinen die Komponenten von einem Ansichtsstatus zum nächsten zu springen. Sie können dieses abrupte Verhalten durch Übergänge unterbinden.

Ein *Übergang* besteht aus einem oder mehreren Effekten, die nacheinander oder simultan abgespielt werden, wenn sich ein Ansichtsstatus ändert. Ein Beispiel: Angenommen, Sie möchten eine Komponente verkleinern, um Platz für eine neue Komponente zu schaffen, wenn die Anwendung von einem Status zu einem anderen wechselt. Sie können einen Übergang definieren, der die erste Komponente langsam minimiert, während eine neue Komponente langsam auf dem Bildschirm erscheint.

Unterstützung für Flex 3-Ansichtsstatus

Flash Builder bietet Unterstützung für die Ansichtsstatus, die in Flex 3 implementiert sind. Wenn Sie ein Projekt erstellen, das das Flex 3 SDK verwendet, kehrt der MXML-Editor sowohl im Designmodus als auch im Quellmodus wieder zur Flex Builder 3-Implementierung zurück. Informationen zum Bearbeiten von Status für das Flex 3 SDK finden Sie in der Dokumentation zu Flex Builder 3.

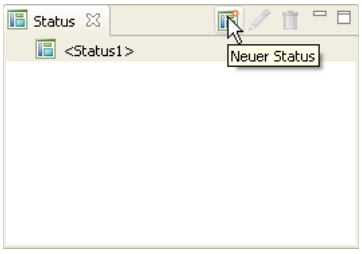
Anzeigestatus erstellen

Anwendungen haben standardmäßig nur einen Ansichtsstatus, den Sie normalerweise als Basisstatus verwenden. Verwenden Sie die Statusansicht von Flash Builder, um zusätzliche Status hinzufügen und das Layout und die Komponenten für jeden Status zu bearbeiten.

1 Entwerfen Sie mit den Layoutwerkzeugen in Flash Builder das Layout des Basisstatus der Anwendung oder Komponente.

Weitere Informationen finden Sie unter „Benutzeroberflächen erstellen“ auf Seite 186.

2 Klicken Sie in der Statusansicht („Fenster“ > „Ansicht anzeigen“ > „Andere“ > „Flash Builder“ > „Status“) in der Symbolleiste auf die Schaltfläche „Neuer Status“.



Das Dialogfeld „Neuer Status“ wird angezeigt.

3 Geben Sie einen Namen für den neuen Status ein.

4 Geben Sie an, ob ein Status, der ein Duplikat eines vorhandenen Status ist, oder ein neuer, leerer Status erstellt werden soll. Klicken Sie auf „OK“.

5 Verwenden Sie die Layoutwerkzeuge in Flash Builder, um den neuen Status zu gestalten.

Sie können Komponenten bearbeiten, verschieben, hinzufügen und löschen. Die vorgenommenen Änderungen für den neuen Status werden Teil des MXML-Codes.

6 Definieren Sie eine Ereignisprozedur, damit der Benutzer zum neuen Status wechseln kann.

Weitere Informationen finden Sie unter „Ansichtsstatus während der Laufzeit ändern“ auf Seite 203.

Nicht-Basisstatus als Anfangs-Ansichtsstatus festlegen

Standardmäßig zeigt eine Anwendung den Basisstatus an, wenn sie aufgerufen wird. Sie können aber auch festlegen, dass beim Starten der Anwendung ein Status als Anfangsstatus angezeigt werden soll.

1 Doppelklicken Sie in der Statusansicht („Fenster“ > „Status“) auf den Ansichtsstatus, der als Anfangsstatus verwendet werden soll.

2 Aktivieren Sie im angezeigten Dialogfeld „Statuseigenschaften bearbeiten“ die Option „Als Anfangsstatus setzen“ und klicken Sie auf „OK“.

Anzeigestatus einer Komponente festlegen

Wenn eine Anwendung mehrere Status hat, kann der Ansichtsstatus einer einzelnen Komponente festgelegt werden.

1 Wählen Sie in der Designansicht des MXML-Editors eine Komponente im Layout aus.

2 Wählen Sie in der Eigenschaftenansicht im Feld „In Status“ die Status aus, in denen die Komponente sichtbar ist.

Ansichtsstatus während der Laufzeit ändern

Wenn Anwendungen ausgeführt werden, müssen Benutzer zwischen Ansichtsstatus wechseln können. Sie können Ereignisprozeduren für Benutzer-Steuerelemente definieren, damit Benutzer bei der Ausführung einer Anwendung zwischen den Status wechseln können.

Die einfachste Methode ist, die Eigenschaft `currentState` zum Klickereignis eines Steuerelements wie z. B. eines Buttons oder eines Links zuzuweisen. Die Eigenschaft `currentState` nimmt den Namen des Ansichtsstatus, den Sie anzeigen möchten, wenn das Klickereignis auftritt. Im Code definieren Sie die Eigenschaft `currentState` wie folgt:

```
click="currentState='viewstatename'"
```

Wenn der Ansichtsstatus für eine bestimmte Komponente definiert ist, müssen Sie auch den Komponentennamen wie folgt angeben:

```
click="currentState='componentID.viewstatename'"
```

Weitere Informationen finden Sie unter Ansichtsstatus.

1 Vergewissern Sie sich, dass der Anfangsstatus ein klickbares Steuerelement wie z. B. einen Button hat.

Wählen Sie im Designmodus des MXML-Editors das gewünschte Steuerelement aus und geben Sie in der Eigenschaftenansicht in das Feld „Beim Klicken“ den folgenden Wert ein:

`currentState='`*`viewstatename`*`'`

viewstatename ist der Name für den Status.

2 Wenn Sie zum Basisstatus wechseln möchten, geben Sie Folgendes ein:

```
currentState=''
```

`''` ist ein leerer String. Er wird durch zwei einzelne Anführungszeichen dargestellt.

3 Um zu testen, ob die Status beim Klicken auf den Button richtig in der Anwendung wechseln, klicken Sie in der Symbolleiste von Flash Builder auf die Schaltfläche „Ausführen“.

Sie können einen Übergang definieren, damit der Wechsel zwischen Ansichtsstatus visuell glatter verläuft. Weitere Informationen finden Sie unter „Übergänge erstellen“ auf Seite 206.

Anzeigestatusgruppen erstellen

Flex bietet Unterstützung für Ansichtsstatusgruppen. Mit dem Attribut `stateGroups` des `<States>`-Tag können Sie ein oder mehrere Status zusammenfassen. Erscheinen beispielsweise in einem Ansichtsstatusset mehrere Komponenten, können Sie eine Ansichtsstatusgruppe erstellen, die alle diese Ansichtsstatus enthält. Wenn Sie anschließend die Eigenschaft `currentState` auf einen der Ansichtsstatus in der Gruppe setzen, werden die Komponenten angezeigt. Weitere Informationen mit Beispielen finden Sie unter Defining view state groups.

Statusgruppen können nicht im Designmodus des MXML-Editors bearbeitet werden. Verwenden Sie den Quellmodus, um Statusgruppen zu erstellen und zu bearbeiten. Der Quellmodus bietet Codehinweise und das Popupmenü „Status anzeigen“, um Ihnen beim Erstellen und Bearbeiten von Statusgruppen zu helfen.

Passen Sie in der Designansicht besonders auf, falls Sie Ansichtsstatusgruppen erstellen. Wenn Sie einen Status in der Designansicht löschen, könnten Sie versehentlich einen Verweis auf eine gelöschte Komponente in einer Statusgruppe hinterlassen.

Anzeigestatus löschen

Sie können in Anwendungen enthaltene Ansichtsstatus in der Regel über die Designansicht des MXML-Editors löschen. Wenn Sie allerdings Statusgruppen erstellt haben, löschen Sie Status über die Quellansicht. Hierdurch wird verhindert, dass versehentlich ein Verweis auf eine Komponente in einem gelöschten Status hinterbleibt.

1. Wählen Sie in der Designansicht des MXML-Editors den Ansichtsstatus aus, der aus der Statusansicht („Fenster“ > „Status“) gelöscht werden soll.
2. Klicken Sie in der Symbolleiste „Statusansicht“ auf die Schaltfläche „Status löschen“.

Mit mehreren Status in einer Anwendung arbeiten

Wenn eine Anwendung mehrere Status enthält, kann über den Designmodus des MXML-Editors die Ansicht für jeden Status so geändert werden, dass nur die für den bestimmten Status definierten Komponenten angezeigt werden. Sie können für jede Komponente den Status angeben, in dem sie sichtbar sein soll.

Komponente eines bestimmten Status bearbeiten

1. Fügen Sie in der Designansicht des Quelleditors über die Statusansicht einer Anwendung einen oder mehrere zusätzliche Status hinzu.
2. Wählen Sie in der Dropdownliste „Status“ die Ansicht des ausgewählten Status aus.
3. Fügen Sie Komponenten zum Status hinzu, bzw. löschen oder ändern Sie im jeweiligen Status enthaltene Komponenten.

   An einem bestimmten Status vorgenommene Änderungen erscheinen nur dann in anderen Status, wenn Sie angeben, dass die jeweilige Komponente in mehreren Status erscheint.

Angeben, dass eine Komponente in mehreren Status erscheint

1. Fügen Sie in der Designansicht des Quelleditors über die Statusansicht einer Anwendung einen oder mehrere zusätzliche Status hinzu.
2. Wählen Sie die gewünschte Komponente in einem Status aus.
3. Wählen Sie in der Eigenschaftenansicht aus, in welchen Status die Komponente erscheint.

   Eine Komponente kann entweder in allen Status, in einem oder in mehreren Status erscheinen.

   Wenn Sie für eine Komponente einen bestimmten Status angeben, wird die Komponente beim Bearbeiten eines anderen Status nicht im Editor angezeigt.

Passen Sie besonders auf, wenn Sie Anwendungen bearbeiten, die mehrere Status enthalten. Komponenten scheinen manchmal zu verschwinden, wenn im Editor zu einem Status gewechselt wird, der eine in einem anderen Status sichtbare Komponente nicht enthält.

Ansichtsstatus im Quellcode erstellen und bearbeiten

Der Quellmodus des MXML-Editors enthält eine Reihe von Funktionen, die beim Bearbeiten des Quellcodes für Ansichtsstatus nützlich sind.

Wenn eine Anwendung Ansichtsstatus deklariert, stellt der MXML-Editor das Popupmenü „Status anzeigen“ an. Wenn Sie im Menü „Status anzeigen“ einen bestimmten Ansichtsstatus wählen, werden Komponenten, die nicht in diesem Status angezeigt werden, nicht hervorgehoben.

Die `includeIn`- und `excludeFrom`-Eigenschaften für MXML-Komponenten geben den Ansichtsstatus oder die Statusgruppe an, in dem bzw. der die Komponente angezeigt wird. Codehinweise im MXML-Editor helfen Ihnen, einen Ansichtsstatus oder eine Statusgruppe für diese Eigenschaften zu wählen.

Sie können auch die Punktschreibweise mit Komponentenattributen verwenden, um einen Ansichtsstatus anzugeben, für den das Attribut gilt. Wenn Sie beispielsweise festlegen möchten, dass eine Button-Komponente in zwei Ansichtsstatus angezeigt wird, und sich die Beschriftung entsprechend des Anzeigestatus ändern soll, verwenden Sie den Punktoperator mit der `label`-Eigenschaft. Codehinweise im MXML-Editor helfen Ihnen, einen Ansichtsstatus zu wählen. Beispiel:

```
<s:Button label.State1="Button in State 1" label.State2="Same Button in State 2">
```

Beispiel für das Arbeiten mit Ansichtsstatus im Quellcode

1 Erstellen Sie eine Anwendung, die mehr als einen Ansichtsstatus enthält.

Fügen Sie im Quellmodus des MXML-Editors nach dem `<s:Application>`-Tag den folgenden Code hinzu:

```
<s:states>
    <s:State name="State1" />
    <s:State name="State2" />
    <s:State name="State3" />
</s:states>
```

Nachdem Sie in der Anwendung die gewünschten Status definiert haben, fügt der MXML-Editor das Popupmenü „Status anzeigen“ hinzu.

2 Fügen Sie im Quellmodus die folgenden Button-Komponenten hinzu:

```
<s:Button includeIn="State1" label="Show State 2"
    click="currentState='State2'" />
<s:Button includeIn="State2" label="Show State 3"
    click="currentState='State3'" />
<s:Button includeIn="State3" label="Show State 1"
    click="currentState='State1'" />

<s:Button
    label.State1="All States: State 1 Label"
    label.State2="All States: State 2 Label"
    label.State3="All States: State 3 Label"
    x="0" y="30"/>
```

In der Standardeinstellung zeigt der Editor Code für alle Status an.

Hinweis: *Die Klickereignisprozeduren für die ersten drei Buttons wechseln durch die Ansichtsstatus.*

3 Bleiben Sie im Quellmodus und wählen Sie im Popupmenü „Status anzeigen“ verschiedene Ansichtsstatus.

Wenn Komponenten im ausgewählten Status nicht sichtbar sind, zeigt der Editor den zugehörigen Code hellgrau an.

Der gesamte Code kann bearbeitet werden, aber das Nichthervorheben von Komponenten, die im gewählten Ansichtsstatus nicht angezeigt werden, hilft, den Code für die verschiedenen Ansichtsstatus zu pflegen.

4 Wechseln Sie zum Designmodus des MXML-Editors.

Wählen Sie entweder mithilfe der Statusansicht oder des Status-Popupmenüs unterschiedliche Ansichtsstatus. Der Editor zeigt die Komponenten entsprechend der für den gewählten Ansichtsstatus definierten Eigenschaften an.

5 Führen Sie die Anwendung aus. Klicken Sie auf die oberste Schaltfläche, um durch die Ansichtsstatus zu wechseln.

Weitere Informationen zum Erstellen und Bearbeiten von Status im Quellcode finden Sie unter Create and apply view states.

Übergänge erstellen

Wenn Sie die Ansichtsstatus in der Anwendung ändern, scheinen die Komponenten von einem Ansichtsstatus zum nächsten zu springen. Sie können diesen Wechsel für die Benutzer visuell glatter gestalten, indem Sie Übergänge verwenden. Ein Übergang besteht aus einem oder mehreren Effekten, die zusammen abgespielt werden, wenn sich ein Ansichtsstatus ändert. Beispielsweise können Sie einen Übergang definieren mit einem Größenänderungseffekt, der eine Komponente im ursprünglichen Ansichtsstatus allmählich minimiert, und einem Einblendeffekt, der eine Komponente allmählich im neuen Ansichtsstatus anzeigt.

1 Erstellen Sie zusätzlich zum Basisstatus mindestens einen weiteren Ansichtsstatus.

2 Definieren Sie in der Quellansicht des MXML-Editors ein Übergangsobjekt, indem Sie ein `<s:transitions>`-Tag und dann ein untergeordnetes `<s:Transition>`-Tag wie im folgenden Beispiel gezeigt schreiben:

```
<s:transitions>
    <mx:Transition id="myTransition">
    </mx:Transition>
</s:transitions>
```

Um mehrere Übergänge zu definieren, fügen Sie zusätzliche untergeordnete `<s:Transition>`-Tags in das `<s:transitions>`-Tag ein.

3 Definieren Sie im `<s:Transition>`-Tag die Änderung im Ansichtsstatus, die den Übergang auslöst. Definieren Sie hierzu die Tag-Eigenschaften `fromState` und `toState` wie im folgenden Beispiel (fettgedruckt) gezeigt:

```
<s:transitions>
    <mx:Transition id="myTransition" fromState="*" toState="checkout">
    </mx:Transition>
</s:transitions>
```

Im Beispiel legen Sie fest, dass der Übergang erfolgen soll, wenn die Anwendung aus einem beliebigen Ansichtsstatus (`fromState="*"`) in den Ansichtsstatus „checkout“ (`toState="checkout"`) wechselt. Der Wert `"*"` ist ein Platzhalterzeichen, der für einen beliebigen Ansichtsstatus steht.

4 Legen Sie im `<mx:Transition>`-Tag fest, ob die Effekte parallel oder nacheinander abgespielt werden sollen. Schreiben Sie hierzu ein untergeordnetes `<mx:Parallel>`- oder `<mx:Sequence>`-Tag wie im folgenden Beispiel (fettgedruckt) gezeigt:

```
<mx:Transition id="myTransition" fromState="*" toState="checkout">
    <mx:Parallel>
    </mx:Parallel>
</mx:Transition>
```

Wenn die Effekte simultan abgespielt werden sollen, verwenden Sie das `<mx:Parallel>`-Tag. Wenn die Effekte nacheinander abgespielt werden sollen, verwenden Sie das `<mx:Sequence>`-Tag.

5 Geben Sie im `<mx:Parallel>`- oder `<mx:Sequence>`-Tag die Komponente(n) an, bei der bzw. bei denen der Übergang erfolgen soll. Setzen Sie hierzu die Eigenschaft `target` (für eine Zielkomponente) oder `targets` (für mehrere Zielkomponenten) auf die ID der Zielkomponente(n) wie im folgenden Beispiel gezeigt:

```
<mx:Parallel targets="{[myVGroup1,myVGroup2,myVGroup3]}">
</mx:Parallel>
```

In diesem Beispiel sind drei VGroup-Container als Zielkomponenten festgelegt. Die Eigenschaft `targets` kann mehrere IDs enthalten.

6 Legen Sie im `<mx:Parallel>`- oder `<mx:Sequence>`-Tag die Effekte fest, die beim Ändern des Ansichtsstatus abgespielt werden sollen. Schreiben Sie hierzu untergeordnete Effekt-Tags wie im folgenden Beispiel (fettgedruckt) gezeigt:

```
<mx:Parallel targets="{[myVBox1,myVBox2,myVBox3]}">
    <mx:Move duration="400"/>
    <mx:Resize duration="400"/>
</mx:Parallel>
```

Eine Liste möglicher Effekte sowie Informationen dazu, wie Sie deren Eigenschaften festlegen, finden Sie unter Introduction to effects.

7 Um den Übergang zu testen, klicken Sie in der Symbolleiste vom Flash Builder auf die Schaltfläche „Ausführen“. Wechseln Sie den Status, nachdem die Anwendung gestartet wurde.

Interaktivität mit Effekten hinzufügen

Ein *Effekt* ist eine optische oder akustische Änderung der Zielkomponente, die über einen in Millisekunden gemessenen Zeitraum erfolgt. Beispiele für Effekte sind Ausblenden, Skalieren (Ändern der Größe) und Verschieben von Komponenten.

Effekte werden als Antwort auf ein Ereignis ausgelöst. Dabei wird das Ereignis oft durch eine Benutzeraktion wie das Klicken auf eine Schaltfläche ausgelöst. Sie können Effekte jedoch auch programmgesteuert initiieren oder als Antwort auf nicht vom Benutzer ausgelöste Ereignisse.

Beispiel: Sie erstellen einen Effekt für eine TextInput-Komponente, der dazu führt, dass die Komponente leicht springt, wenn der Benutzer durch Drücken der Tabulatortaste zu dieser Komponente wechselt. Oder: Sie definieren einen Effekt für eine Label-Komponente, durch die die Beschriftung ausgeblendet wird, wenn der Benutzer mit der Maus über sie fährt.

In Flash Builder können Sie Effekte als Eigenschaft einer MXML-Komponente definieren. Implementieren Sie den Effekt in der Quellansicht des MXML-Editors.

Die Effekte werden bei Spark- und MX-Komponenten unterschiedlich implementiert. Informationen zum Erstellen von Effekten in MXML- und ActionScript-Code finden Sie unter Introduction to effects.

Effekte für eine Komponente erstellen

Normalerweise werden Effekte im Quellmodus des MXML-Editors definiert. Effekte werden häufig von der Ereignisprozedur einer Komponente aufgerufen. Beispiel: Um von einer Schaltfläche aus einen Effekt aufzurufen, verwenden Sie die Klick-Ereignisprozedur. Siehe Applying effects.

In Flash Builder können Sie jedoch für eine Eigenschaft einer MXML-Komponente einen Effekt definieren.

1 Klicken Sie im Designmodus des MXML-Editors im Designbereich auf eine Komponente.

2 Definieren Sie die Effekteigenschaft für Spark-Komponenten:

a Wählen Sie in der Eigenschaftenansicht das Symbol für die Kategorieansicht.

b Wählen Sie in der Kategorie „Effekte“ eine Eigenschaft und geben Sie einen Effekt an.

Beispiel: Für die rollOverEffect-Eigenschaft einer Schaltfläche können Sie den Fade-Effekt angeben. Eine Liste der verfügbaren Effekte finden Sie unter Available effects.

3 Speichern Sie und führen Sie die Datei aus, um den Effekt anzuzeigen.

Benutzeroberflächen mithilfe von Skins modifizieren

Skinklassen modifizieren das Aussehen von Steuerelementen auf einer Benutzeroberfläche. Die Methoden zum Erstellen, Bearbeiten und Importieren von Skins sind bei Spark-Komponenten und MX-Komponenten verschieden.

Spark-Skins

Spark-Skins steuern alle visuellen Elemente einer Komponente, einschließlich des Layouts. Spark-Skins können mehrere Elemente enthalten, wie z. B. grafische Elemente, Text, Bilder und Übergänge. Spark-Skins unterstützen Status. Mithilfe von Skins können Sie das Aussehen eines jeden einzelnen Status von Komponenten definieren. Normalerweise geben Skins Mindestanforderungen für die Größe der Komponente an. Informationen zur Implementierung von Spark-Skins in Flex finden Sie unter Spark-Skins.

Mit Flash Builder können Sie Skins für eine Spark-Komponente generieren und bearbeiten. Wenn Flash Builder eine Skin generiert, wird eine Skinklasse in MXML erstellt. Im MXML-Editor können Sie das Aussehen modifizieren, das von der Skin definiert ist. Einige Änderungen können Sie im Designmodus des Editors vornehmen, andere erfordern das Bearbeiten der MXML-Datei im Quellmodus. Siehe „Skins für Spark-Komponenten generieren und bearbeiten“ auf Seite 208.

Skins für MX-Komponenten

Skins für MX-Komponenten können entweder Bitmap-Grafiken oder Vektorgrafiken sein. Eine Bitmap-Grafik (eine grafische Skin) besteht aus einzelnen Pixeln, die zusammen ein Bild ergeben. Eine Vektorgrafik (eine programmgesteuerte Skin) besteht aus einem Satz von Liniendefinitionen, die den Anfangs- und Endpunkt einer Linie sowie die Stärke, die Farbe und andere von Adobe® Flash® Player zum Zeichnen der Linie benötigte Informationen angeben. Informationen zur Implementierung von Skins für MX-Komponenten in Flex finden Sie unter About MX component skinning.

Sie können Flash Builder zum Importieren von Skingrafiken für MX-Komponenten verwenden. Siehe „Skins für MX-Komponenten importieren“ auf Seite 212.

Das `mx.skins.spark`-Paket definiert Spark-Skins für MX-Komponenten.

Skins für Spark-Komponenten generieren und bearbeiten

Mit Flash Builder können Sie Skins für Spark-Komponenten generieren und bearbeiten. Wenn Sie eine Skin generieren, verwendet Flash Builder eine Skin des Projektdesigns. Das standardmäßige Projektdesign ist Spark. Sie können das Projektdesign im Aussehen-Bedienfeld ändern. Siehe „Designs übernehmen“ auf Seite 191.

Beim Generieren einer Skin für eine Komponente erstellt Flash Builder eine MXML-Datei, die die Skinklasse für die Komponente implementiert. Sie können angeben, ob die Skinklasse als Kopie einer vorhandenen Skin generiert oder ob eine leere Skinklassendatei erzeugt werden soll.

Verwenden Sie den MXML-Editor im Designmodus und im Quellmodus, um die Skin zu bearbeiten. Wählen Sie im Designmodus mithilfe der Gliederungsansicht die Elemente der Skin aus, die Sie bearbeiten möchten. Navigieren Sie in der Statusansicht zwischen den Status der Komponente. Einige Teile einer Skin können nicht im Designmodus bearbeitet werden. Nutzen Sie den Quellmodus, um Teile von Skins zu bearbeiten, die im Designmodus nicht verfügbar sind.

Einige Komponenten enthalten Unterkomponenten. Beispielsweise enthält eine HSlider-Komponente Button-Komponenten, die den Schieberegler und die Schiebereglerleiste der HSlider-Komponente definieren. Unterkomponenten können ausschließlich im Quellmodus mit einer Skin versehen werden.

Komponentenstatus, Skinteile und Hostkomponenten

Skins definieren das Aussehen einer Komponente für jeden ihrer Status. Beispielsweise hat ein Spark-Button die vier Status „normal“, „darüber“, „gedrückt“ und „deaktiviert“. Wenn Sie eine Skin für einen Spark-Button generieren, können Sie das Aussehen für jeden dieser Status angeben.

Jede Komponente verfügt über Teile, die angepasst werden können. Bei einer Button-Komponente können Sie beispielsweise die Füllfarbe der Schaltfläche, die Textattribute für die Button-Beschriftung und die Rect-Komponenten, die den Rahmen des Buttons ausmachen, gestalten.

Wenn Sie mit Flash Builder Skins für eine Komponente erstellen, geben Sie eine Hostkomponente an, auf der die generierte Skin basiert. Standardmäßig ist die Hostkomponente die Basisklasse der Komponente, die mit einer Skin versehen wird. Sie können jedoch eine andere Hostkomponente wählen.

Hinweis: *Das Angeben einer Hostkomponente für eine Skinklasse ist erforderlich, wenn mit Flash Builder Skinklassen generiert werden. Wenn Sie jedoch Skinklassen direkt im Quellcode erstellen, ist eine Hostkomponente nicht erforderlich.*

Skinning-Vertrag zwischen einer Skin und ihrer Hostkomponente

Der Skinning-Vertrag zwischen einer Skinklasse und einer Komponentenklasse definiert die Regeln, die alle Member einhalten müssen, damit sie miteinander kommunizieren können.

Die Skinklasse muss Skinstatus deklarieren und das Aussehen von Skinteilen definieren. Skinklassen geben normalerweise auch die Hostkomponente an und binden manchmal an Daten, die in der Hostkomponente definiert sind.

Die Komponentenklasse deklariert, welche Skinklasse sie verwendet. Außerdem muss sie Skinstatus und Skinteile mit Metadaten identifizieren. Wenn die Skinklasse an Daten in der Hostkomponente gebunden wird, muss die Hostkomponente diese Daten definieren.

Die folgende Tabelle zeigt diese Regeln des Skinning-Vertrags:

	Skinklasse	**Hostkomponente**	**Erforderlich?**

Hostkomponente	`<fx:Metadata>`
	`[HostComponent("spark.components.Button")]`
	`</fx:Metadata>`
		–	Nein

Skinstatus	`<s:states>`
	`<s:State name="up"/>`
	`</s:states>`
		`[SkinStates("up")];`
		`public class Button {`
		`...`
		`}`
			Ja

Skinteile	`<s:Button id="upButton"/>`	`[SkinPart(required="false")]` `public var:upButton Button;`	Ja
Daten	`text="{hostComponent.title}"`	`[Bindable]` `public var title:String;`	Nein

Skindeklaration

In Flash Builder ist eine Skindeklaration die Datei, die die Skin für eine Komponente implementiert. Flex definiert für jede visuelle Komponente eine Skindeklaration. Wenn Sie für eine Komponente eine neue Skin generieren, generiert Flash Builder die Skindeklaration.

Sie können die Skindeklaration für ausgewählte Komponenten anzeigen:

1 Wählen Sie im Designmodus des MXML-Editors eine Spark-Komponente im Designbereich aus.

2 Wählen Sie im Kontextmenü der Komponente „Skindeklaration öffnen“.

Die Klasse, die die Skin implementiert, wird im Quellmodus des Editors geöffnet.

Wenn Sie die Klasse selbst erstellt haben, können Sie die Datei bearbeiten.

Zum Öffnen einer Skindeklarationsdatei können Sie auch folgendermaßen vorgehen:

- Wählen Sie die Komponente aus und klicken Sie in der Eigenschaftenansicht im Bereich „Stil“ auf das Symbol neben dem Feld „Skin“.
- Wechseln Sie in den Quellmodus und wählen Sie bei ausgewählter Spark-Komponente im Flash Builder-Menü „Navigieren“ > „Skindeklaration öffnen“.

Skins für Spark-Buttons generieren und bearbeiten (Designmodus)

Dieses Beispiel generiert eine Skinklasse für einen Spark-Button und zeigt, wie Sie eine Kombination von Flash Builder-Ansichten benutzen, um die Skin zu bearbeiten. Es setzt voraus, dass Sie in einem Flex-Projekt unter Verwendung des standardmäßigen Spark-Designs arbeiten.

1 Erstellen einer Anwendungsdatei Fügen Sie im Designmodus des Editors der Anwendung einen Spark-Button hinzu.

2 Wählen Sie im Kontextmenü des Button den Befehl „Skin erstellen“.

Das Dialogfeld „MXML-Skin - Neu“ wird angezeigt.

Zum Öffnen des Dialogfeldes „MXML-Skin - Neu“ können Sie auch folgendermaßen vorgehen:

- Klicken Sie in der Eigenschaftenansicht im Bereich „Stil“ auf das Symbol neben dem Feld „Skin“.
- Wählen Sie im Flash Builder-Menü „Neu“ > „MXML-Skin“.
- Wählen Sie im Designmodus des MXML-Editors die Optionen „Design“ > „Skin erstellen“.

3 Geben Sie im Dialogfeld „MXML-Skin - Neu“ Folgendes an:

- Quellordner und Paket für die generierte Skindeklaration.
- Name

  Den Namen für die Skinklasse, die Sie erstellen.

- Hostkomponente

  Wenn Sie die Standardkomponente ändern möchten, klicken Sie auf „Durchsuchen“ und wählen Sie eine Hostkomponente.

- (Empfohlen) Aktivieren Sie das Kontrollkästchen „Erstellen als Kopie von“ und aktivieren Sie nicht das Kontrollkästchen „ActionScript-Stilcode entfernen“.

  Wenn Sie mit Skinning nicht vertraut sind, verwenden Sie zum Erstellen einer Skinklasse eine Kopie. Modifizieren Sie den ActionScript-Stilcode.

- (Fortgeschrittene Benutzer) Wenn Sie mit den Schritten zum Erstellen von Skinklassen vertraut sind, gehen Sie folgendermaßen vor:

  Aktivieren Sie die Option zum Entfernen von ActionScript-Stilcode oder deaktivieren Sie die Option zum Erstellen einer Kopie einer vorhandenen Klasse.

  Wenn Sie keine Kopie einer vorhandenen Klasse erstellen, generiert Flash Builder eine leere Skinklassendatei mit einigen nützlichen Hinweisen.

  Die verbleibenden Schritte dieses Vorgangs setzen voraus, dass Sie die empfohlene Option zum Generieren einer Skinklasse gewählt haben.

4 Klicken Sie auf „Fertig stellen“.

Flash Builder generiert eine Skinklassendatei und öffnet sie im Designmodus des MXML-Editors.

Die Button-Komponente wird gewählt.

In der Statusansicht wird der Status „normal“ des Buttons gewählt.

5 Modifizieren Sie für jeden Status des Buttons die Textformate, Inhaltshintergrundstile und Farbstile.

Nehmen Sie mithilfe der Bearbeitungswerkzeuge im Bereich „Stil“ der Eigenschaftenansicht die erforderlichen Änderungen vor.

6 Öffnen Sie die Gliederungsansicht:

Der Knoten auf der obersten Ebene (SparkSkin) ist ausgewählt.

7 Wählen Sie in der Gliederungsansicht „Rect shadow“, um Stile für den Schatten des Buttons zu definieren.

Die Werkzeuge im Bereich „Stil“ sind nicht verfügbar.

8 Wechseln Sie in den Quellmodus des Editors.

Flash Builder hebt die Rect-Komponente hervor, die den Schatten des Buttons definiert. Nehmen Sie Änderungen am Schatten des Buttons vor.

9 Speichern Sie die Skinklassendatei und Ihre Anwendungsdatei.

Im Designmodus des MXML-Editors können Sie die Skin für den Button anzeigen, sofern Sie die empfohlene Option in Schritt 3 gewählt haben. Wenn die Stile nicht angezeigt werden, klicken Sie auf das Symbol zum Aktualisieren des Designbereichs.

Die Anwendungsdatei hat einen Verweis auf die von Ihnen erstellte Skinklasse hinzugefügt.

10 Führen Sie die Anwendung aus, um zu sehen, wie sich die Skin bei den Button-Status „normal“, „darüber“, „gedrückt“ und „deaktiviert“ ändert.

Skins für Spark-Komponenten erstellen und bearbeiten (Quellmodus)

Sie können das Dialogfeld „MXML-Skin - Neu“ direkt im Quellmodus des Editors öffnen. Beispielsweise können Sie folgendermaßen vorgehen, um eine skinClass für eine Spark-Button-Komponente zu erstellen.

1 Klicken Sie im Quellmodus des Editors in ein `<s:Button>`-Tag und geben Sie Folgendes ein:

```
<s:Button skinClass="
```

Nachdem Sie das erste Anführungszeichen für den skinClass-Namen eingegeben haben, wird ein Kontextmenü angezeigt.

2 Wählen Sie in den Codehinweisen den Eintrag „Skin erstellen“ und öffnen Sie durch Drücken der Eingabetaste das Dialogfeld „MXML-Skin - Neu“.

Dieses Dialogfeld ist identisch mit dem Dialogfeld, das im Designmodus geöffnet wird.

Informationen zum Erstellen der skinClass finden Sie im Abschnitt „Skins für Spark-Buttons generieren und bearbeiten (Designmodus)“ auf Seite 210.

3 Klicken Sie auf „Fertig stellen“.

Flash Builder generiert aufgrund der im Dialogfeld „MXML-Skin - Neu“ gewählten Optionen eine neue skinClass. Der Editor wechselt zum Quellcode für die neu erstellte Klasse.

4 Bearbeiten Sie die skinClass.

Speichern Sie die Klassendatei und Ihre Anwendungsdatei.

Hinweis: *Sie können die generierte Skinklasse in CSS konvertieren, um die angewendeten Stile anzuzeigen. Siehe „Skins in einen CSS-Stil konvertieren“ auf Seite 212.*

Skins in einen CSS-Stil konvertieren

Mit Flash Builder können Sie die Skin einer Komponente in einen CSS-Stil konvertieren. Der Vorteil der Konvertierung der Skin in einen Stil besteht darin, dass der Stil als Typselektor für alle Komponenten dieser Klasse verwendet werden kann. Legen Sie andernfalls die skinClass-Eigenschaft für jede Komponente fest.

Die folgende Prozedur zeigt, wie Sie eine Skin für einen Spark-Button in einen CSS-Stil konvertieren.

1 Generieren und bearbeiten Sie eine Skin für eine Button-Komponente.

2 Wählen Sie im Designmodus des Editors den Button. Klicken Sie in der Eigenschaftenansicht im Bereich „Stil“ auf „In CSS konvertieren“.

3 Wählen oder erstellen Sie im Dialogfeld „Neue Stilregel“ eine CSS-Datei für den Stil.

   Wenn im Projekt, das Sie verwenden möchten, keine CSS-Datei vorhanden ist, klicken Sie auf „Neu“, um eine Datei zu erstellen.

4 Geben Sie den Selektortyp an. Wählen Sie aus Folgendem:

   - Alle Komponenten

     Der Stil wird auf alle Komponenten in der Anwendung angewendet.

   - Alle Komponenten mit Stilnamen

     Komponenten geben diese Stilauswahl nach Namen an. Wenn Sie diese Option wählen, geben Sie einen Namen für den Typselektor an.

   - Bestimmte Komponente

     Der Stil gilt nur für die gewählte Komponente.

   - Bestimmte Komponente mit Stilnamen

     Der Stil gilt nur für die gewählte Komponente und verweist über den Namen des Typselektors auf den Stil. Wenn Sie diese Option wählen, geben Sie einen Namen für den Typselektor an.

5 Nachdem Sie einen Selektortyp gewählt haben, klicken Sie auf „OK“.

   Flash Builder generiert oder aktualisiert die angegebene CSS-Datei. Flash Builder ändert auch den Quellcode in der Anwendung, um auf den Typselektor in der CSS-Datei zu verweisen.

   Flash Builder entfernt Verweise auf die `skinClass`-Eigenschaft für die Komponente.

Skins für MX-Komponenten importieren

Mit dem Assistenten zum Importieren von Skins können Sie sowohl Vektor- als auch Bitmapgrafiken aus den CS5-Versionen von Flash Professional, Fireworks, Illustrator und Photoshop importieren. (Als Bitmaps können PNG-, JPG- und GIF-Dateien verwendet werden.) Die Grafiken können anschließend als Skins für Flex-Komponenten verwendet werden.

Hinweis: *Adobe stellt mehrere Skin-Vorlagen zur Verfügung, um das Erstellen von Skins für die eingebauten Flex-Komponenten zu erleichtern. Verwenden Sie die Vorlagen in Flash, Fireworks, Illustrator oder Photoshop, um die Grafik zu erstellen. Sie können auch Flash verwenden, um voll funktionsfähige benutzerdefinierte Flex-Komponenten zu erstellen. Weitere Informationen finden Sie in den Artikeln unter Importieren von Flash Professional-Elementen in Flex.*

1 Wählen Sie „Datei“ > „Importieren“ > „Flash Builder“ > „Skin-Bildmaterial“.

2 Im Dialogfeld „Skins importieren“:

- Wählen Sie einen Ordner mit Bitmaps oder eine SWC- oder SWF-Datei, aus der Skins importiert werden sollen, oder klicken Sie auf „Durchsuchen“ und wählen Sie den Ordner bzw. die Datei aus. Folgende Dateitypen werden unterstützt:
  - In Adobe Flash Professional CS5 erstellte AS3-SWF- und AS3-SWC-Dateien
  - Vektorgrafiken, die in Adobe Illustrator® erstellt und als SWF-Dateien für Flash Player 8 exportiert wurden
  - Bitmaps in den Formaten PNG, GIF und JPG.
- Wählen Sie einen Ordner, in den die Skins importiert werden sollen. Der Ordner muss ein Quellordner für ein Flex-Projekt sein (Sie können auch einen Unterordner im Quellordner angeben). Standardmäßig ist der Ordner für das Flex-Projekt ausgewählt, das momentan geöffnet ist.
- Der Name des Standardordners im Feld „Grafiken in Unterordner kopieren“ basiert auf dem Ordner oder den Elementen, die importiert werden. Klicken Sie auf „Durchsuchen“, um einen anderen Ordner auszuwählen.
- Geben Sie im Feld „Stilregeln für Skins erstellen in“ einen Namen für eine CSS-Datei ein, die die Stilregeln enthalten soll. Der Standardname basiert auf dem Namen des Grafikordners oder der FLA-Datei, der bzw. die importiert wird.
- Aktivieren Sie die Option „Alle vorhandenen Regeln in Datei löschen“, wenn die angegebene CSS-Datei beim Importieren überschrieben werden soll (anstatt die Skins zu importieren und andere vorhandene Definitionen in der CSS-Datei beizubehalten). Die Option ist standardmäßig deaktiviert und wenn die CSS-Datei nicht existiert, ist sie abgeblendet.
- Im Feld „Stile auf Anwendung anwenden“ wird standardmäßig die im Flex-Navigator oder in der aktiven Editoransicht ausgewählte Datei oder die Hauptanwendungsdatei für das Projekt angezeigt.
- Klicken Sie auf „Weiter“.

3 Wählen Sie im nächsten Dialogfeld („Skins importieren“) die Skins aus, die Sie importieren möchten. Geben Sie außerdem an, welcher CSS-Stiltyp und welche Skin-Teil-Eigenschaft verwendet werden soll. Sie können Elemente entweder einzeln auswählen oder auf „Alle auswählen“ bzw. „Auswahl aufheben“ klicken.

- Wenn Elemente keinen gültigen Stil- oder Skin-Teil-Eigenschaftennamen haben, sind sie standardmäßig nicht ausgewählt. Die folgenden Beispiele zeigen die in Flash Builder verwendete Namenskonvention:
  - Button_upSkin;
  - Button_glow_downSkin (zur Eigenschaft „downSkin“ der Stilregel „Button.glow“ zugeordnet);
  - TabBar-tab_upSkin (Eigenschaft „upSkin“ ist zur Eigenschaft „tabStyleName“ der Stilregel „TabBar“ zugeordnet);
  - MeinebenutzerdefinierteKomponente_borderSkin.

  Bei benutzerdefinierten Komponenten ist das Element ausgewählt, wenn es an einer Stelle im Projekt definiert ist, in das Sie das Element importieren.
- Wählen Sie (sofern erforderlich) in jeder Spalte einen Stil und einen Skin-Teil für die Popupmenüs.
- Klicken Sie auf „Fertig stellen“.

  Eine CSS-Datei wird erstellt und in der Quellansicht angezeigt. Die CSS-Datei wird an die im Assistenten angegebene Anwendung angehängt. Wenn Sie eine SWC-Datei importieren, wird diese automatisch zum Bibliothekspfad des Projekts hinzugefügt.

Benutzerdefinierte Elementrenderer generieren

Sie können Elementrenderer für listenbasierte Steuerelemente für Desktop- und Mobilanwendungen erstellen.

Bei Desktopanwendungen können Sie benutzerdefinierte Elementrenderer für listenbasierte Spark-Steuerelemente wie List und ComboBox erstellen. Spark-Elementrenderer können auch mit einigen MX-Steuerelementen verwendet werden, wie z. B. mit den MX DataGrid- und MX Tree-Steuerelementen.

Bei Mobilanwendungen können Sie benutzerdefinierte Elementrenderer für listenbasierte Steuerelemente für Mobilanwendungen erstellen.

Mit benutzerdefinierten Elementrenderern können Sie die Anzeige eines Datenelements in einer DataGroup, einem SkinnableDataContainer oder einer Unterklasse dieser Container steuern. Das von einem Elementrenderer definierte Aussehen kann die Schriftart, Hintergrundfarbe, Rahmen und andere visuelle Aspekte des Datenelements umfassen. Ein Elementrenderer definiert auch das Aussehen eines Datenelements, wenn der Benutzer mit ihm interagiert. Beispielsweise kann der Elementrenderer das Datenelement auf eine bestimmte Art anzeigen, wenn der Benutzer den Mauszeiger über das Datenelement bewegt. Es wird jedoch auf andere Weise angezeigt, wenn der Benutzer das Datenelement wählt, indem er darauf klickt.

Mit Flash Builder können Sie Elementrenderer generieren und bearbeiten. Wenn Flash Builder Elementrenderer für eine Desktopanwendung generiert, wird eine der folgenden Vorlagen verwendet:

- Spark-Komponenten

  Nutzen Sie diese Vorlage für listenbasierte Spark-Steuerelemente, wie z. B. List und ComboBox.

- MX Advanced DataGrid
- MX DataGrid
- MX Tree

Wenn Flash Builder Elementrenderer für listenbasierte Steuerelemente in einer Mobilanwendung erstellt, wird eine der folgenden Vorlagen verwendet:

- Symbolelementrenderer

  Verwenden Sie diese Vorlage, um einen benutzerdefinierten Elementrenderer für listenbasierte Steuerelemente zu erstellen, indem Sie die Eigenschaften für die Beschriftung, die Meldung und das Symbol festlegen. Die Elementrendererdatei wird in MXML erstellt.

  Weitere Informationen finden Sie unter „Elementrenderer für Mobilanwendungen erstellen und bearbeiten (Designmodus)“ auf Seite 216.

- Benutzerdefinierter ActionScript-Elementrenderer

  Verwenden Sie diese Vorlage zum Erstellen eines einfachen Elementrenderers für listenbasierte Steuerelemente. Sie können den Elementrenderer anschließend anpassen, um das Aussehen der Liste nach Bedarf anzupassen. Die Elementrendererdatei wird in ActionScript erstellt.

Sie können den Assistenten für neue Elementrenderer sowohl im Design- als auch im Quellmodus des MXML-Editors öffnen. In diesem Assistenten geben Sie einen Namen und eine Vorlage für den Elementrenderer an. Flash Builder generiert eine MXML- oder ActionScript-Datei, die den Elementrenderer implementiert.

Komponenten in der Anwendung verweisen mithilfe der `itemRenderer`-Eigenschaft auf den generierten Elementrenderer.

Weitere Informationen zum Erstellen und Verwenden von Elementrenderern finden Sie unter Benutzerdefinierte Spark-Elementrenderer.

Hier finden Sie Informationen zum Erstellen von Spark Item-Renderern von Adobe Flash Builder-Ingenieur Balaji Sridhar.

Elementrenderer für eine MX Tree-Komponente generieren und bearbeiten (Designmodus)

Dieses Beispiel generiert einen Elementrenderer für eine MX Tree-Komponente und erläutert, wie Sie mithilfe einer Kombination von Flash Builder-Ansichten den Elementrenderer bearbeiten. Es setzt voraus, dass Sie in einem Flex-Projekt unter Verwendung des standardmäßigen Spark-Designs arbeiten.

1 Erstellen einer Anwendungsdatei Fügen Sie im Designmodus des Editors der Anwendung eine MX Tree-Komponente hinzu.

   Füllen Sie das Tree-Element mit Daten, die angezeigt werden können, wenn die Anwendung ausgeführt wird.

2 Wählen Sie im Tree-Kontextmenü den Befehl „Elementrenderer erstellen“.

   Das Dialogfeld „Neuer Elementrenderer“ wird angezeigt.

   Zum Öffnen des Dialogfelds „Neuer Elementrenderer“ können Sie auch folgendermaßen vorgehen:

   - Klicken Sie in der Eigenschaftenansicht im Bereich „Allgemein“ auf das Symbol neben dem Feld „Elementrenderer“.
   - Wählen Sie im Flash Builder-Menü „Neu“ > „Elementrenderer“.

3 Geben Sie im Dialogfeld „Neuer Elementrenderer“ Folgendes an:

   - Quellordner und Paket für die generierte Elementrendererdeklaration.
   - Name

     Name der Elementrendererklasse, die Sie erstellen.

   - Vorlage

     Wählen Sie die Vorlage aus, die beim Generieren des Elementrenderers verwendet werden soll.

4 Klicken Sie auf „Fertig stellen“.

   Flash Builder generiert eine ItemRenderer-Klassendatei und öffnet sie im Designmodus des MXML-Editors.

   Die ItemRenderer-Komponente wird gewählt.

   In der Statusansicht wird der normale Status des Tree-Elements gewählt.

5 Modifizieren Sie das Aussehen für jeden Tree-Status in der generierten ItemRenderer-Klasse.

   a Öffnen Sie die Gliederungsansicht:

   Beachten Sie, dass der Knoten auf der obersten Ebene, MXTreeItemRenderer, gewählt ist.

   Modifizieren Sie in der Eigenschaftenansicht im Bereich „Stil“ das Aussehen der Tree-Elemente.

   b Wählen Sie in der Gliederungsansicht andere Komponenten des MXTreeItemRenderer, um das Aussehen dieser Komponenten zu modifizieren.

   Beachten Sie, dass die Werkzeuge im Bereich „Stil“ nicht immer verfügbar sind.

   Wenn die Werkzeuge im Bereich „Stil“ nicht verfügbar sind, können Sie das Aussehen mithilfe des Quellmodus des Editors definieren. Wenn Sie in den Quellmodus wechseln, wird der Quellcode der in der Gliederungsansicht ausgewählten Komponente hervorgehoben.

6 Führen Sie die Anwendung aus, um sich einen Eindruck zu verschaffen, wie der ItemRenderer das Aussehen des Tree ändert.

Elementrenderer für Mobilanwendungen erstellen und bearbeiten (Designmodus)

In diesem Beispiel wird ein listenbasierter Elementrenderer für Mobilkomponenten erstellt. Dabei wird angenommen, dass Sie in einem Flex-Mobilprojekt unter Verwendung des standardmäßigen Spark-Designs arbeiten.

1 Erstellen Sie eine Mobilanwendungsdatei. Fügen Sie im Designmodus des Editors der Anwendung Komponenten hinzu.

2 Wählen Sie im Kontextmenü für die Mobilanwendungsdatei „Neu“ > „Elementrenderer“. Sie können auch in der Standardansicht des Eigenschafteninspektors auf „Elementrenderer erstellen“ klicken.

Das Dialogfeld „Neuer Elementrenderer“ wird angezeigt.

3 Geben Sie im Dialogfeld „Neuer Elementrenderer“ Folgendes an:

- Quellordner und Paket für die generierte Elementrendererdeklaration.
- Name

  Name der Elementrendererklasse, die Sie erstellen.
- Vorlage

  Wählen Sie die Vorlage für den Symbol-Elementrenderer für Listen in Mobilanwendungen.
- Beschriftungsfeld

  Der Name des Felds in den Daten, das als Beschriftung angezeigt werden soll.
- Meldungsfeld

  Der Name des Felds in den Daten, das als Inhalt der Meldung angezeigt werden soll.
- Symbolfeld

  Der Name des Felds in den Daten, der den Pfad zum Symbol enthält.
- Symbolbreite

  Die Breite des Symbols. Der Standardwert ist 64 Pixel.
- Symbolhöhe

  Die Höhe des Symbols. Der Standardwert ist 64 Pixel.
- Dekorierungsklasse

  Wählen Sie ein Bild in einem der Formate GIF, JPEG oder PNG. Das ausgewählte Bild wird in die Anwendung eingebettet. Beispiel: `decoratorClass="@Embed('/foo/myfoo.png')"`

  Sie können auch eine FXP-Datei wählen. Um auf eine FXP-Datei zu verweisen, fügen Sie den Pfad zu dieser Datei hinzu. Beispiel: `decoratorClass="{assets.Chevron}"`

4 Klicken Sie auf „Fertig stellen“.

Flash Builder generiert eine ItemRenderer-Klassendatei und öffnet sie im Designmodus des MXML-Editors.

5 Führen Sie die Anwendung aus, um die Funktionsfähigkeit des ItemRenderer zu testen.

Elementrenderer (Quellmodus) erstellen und bearbeiten

Sie können das Dialogfeld „Neuer Elementrenderer“ direkt im Quellmodus des Editors öffnen. Beispielsweise können Sie folgendermaßen vorgehen, um einen Elementrenderer für eine Spark-List-Komponente zu erstellen.

1 Klicken Sie im Quellmodus des Editors in ein `<s:List>`-Tag und geben Sie Folgendes ein:

```
<s:List itemRender="
```

Nachdem Sie das erste Anführungszeichen für den Elementrenderer-Klassennamen eingegeben haben, wird ein Kontextmenü angezeigt.

2 Doppelklicken Sie auf „Elementrenderer erstellen“, um das Dialogfeld „Neuer Elementrenderer“ zu öffnen.

Dieses Dialogfeld ist identisch mit dem Dialogfeld, das im Designmodus geöffnet wird.

Anweisungen zum Erstellen des Elementrenderers für Desktopanwendungen finden Sie im Abschnitt „Elementrenderer für eine MX Tree-Komponente generieren und bearbeiten (Designmodus)“ auf Seite 215.

Anweisungen zum Erstellen des Elementrenderers für Mobilanwendungen finden Sie unter „Elementrenderer für Mobilanwendungen erstellen und bearbeiten (Designmodus)“ auf Seite 216.

3 Klicken Sie auf „Fertig stellen“.

Flash Builder generiert basierend auf den im Dialogfeld „Neuer Elementrenderer“ gewählten Optionen einen neuen Elementrenderer. Der Editor wechselt zum Quellcode für die neu erstellte Klasse.

4 Bearbeiten Sie die Elementrendererklasse.

Speichern Sie die Klassendatei und Ihre Anwendungsdatei.

ItemRenderer-Deklaration

In Flash Builder ist eine ItemRenderer-Deklaration die Datei, die den benutzerdefinierten ItemRenderer für eine Komponente implementiert.

Sie können die benutzerdefinierte ItemRenderer-Deklaration für ausgewählte Komponenten anzeigen:

1 Wählen Sie im Designmodus des MXML-Editors eine Komponente aus, für die Sie einen benutzerdefinierten Elementrenderer implementiert haben.

2 Wählen Sie im Kontextmenü der Komponente „Elementrendererdeklaration öffnen“.

Die Klasse, die den Elementrenderer implementiert, wird im Quellmodus des Editors geöffnet. Zum Öffnen der Elementrendererdeklaration können Sie auch folgendermaßen vorgehen:

- Wählen Sie im Designmodus die Komponente aus. Klicken Sie in der Eigenschaftenansicht im Bereich „Allgemein“ auf das Symbol neben dem Feld „Elementrenderer“.
- Wechseln Sie in den Quellmodus und wählen Sie bei ausgewählter Komponente im Flash Builder-Menü „Navigieren“ > „Skindeklaration öffnen“.

Steuerelemente an Daten binden

Für den Fall, dass auf einen Datendienst zugegriffen wird, bietet Flash Builder Werkzeuge zum Binden von Daten an Datensteuerelemente wie z. B. ein DataGrid an. Flash Builder erstellt die Spalten eines DataGrid auf Grund der vom Dienst zurückgegebenen Daten. Normalerweise müssen Sie die generierten DataGrid-Spalten konfigurieren. Flash Builder bietet einen Editor zum Konfigurieren von Spalten einer DataGrid- und AdvancedDataGrid-Komponente.

Weitere Informationen zum Binden von Daten an Datensteuerelemente finden Sie unter Binden von Dienstmethoden an Steuerelemente.

DataGrid- und AdvancedDataGrid-Komponenten konfigurieren

Sie können DataGrid-Spalten im Designmodus des MXML-Editors konfigurieren. Die folgende Prozedur zeigt, wie Sie die Spalten einer DataGrid-Komponente konfigurieren, die auf einen Datendienst zugreift. Auf ähnliche Weise können Sie die Spalten einer AdvancedDataGrid-Komponente konfigurieren.

DataGrid-Spalten konfigurieren

1 Fügen Sie im Designmodus des MXML-Editors ein DataGrid- (oder AdvancedDataGrid)-Steuerelement hinzu. Binden Sie das Steuerelement an Daten, die von einem Datendienst zurückgegeben wurden.

Weitere Informationen finden Sie unter Binden von Dienstmethoden an Steuerelemente.

2 Wählen Sie das DataGrid und wählen Sie dann im Eigenschafteninspektor „Spalten konfigurieren“.

Sie können „Spalten konfigurieren“ auch im Kontextmenü des DataGrid wählen.

3 Fügen Sie im Dialogfeld „Spalten konfigurieren“ über die Schaltflächen „Hinzufügen“, „Löschen“, „Nach oben“ bzw. „Nach unten“ Spalten hinzu bzw. entfernen Sie Spalten oder ändern Sie deren Reihenfolge.

4 Bearbeiten Sie häufig verwendete Eigenschaften einer ausgewählten Spalte in der Standardansicht des Dialogfelds „Spalten konfigurieren“.

- Datenbindung

  Wählen Sie das Datenfeld, das in der Spalte angezeigt werden soll. Das Kombinationsfeld „An Feld binden“ zeigt alle verfügbaren Felder von den zurückgegebenen Daten an. Wenn das DataGrid bearbeitbar ist, können Sie festlegen, ob Daten in dieser Spalte bearbeitbar sein sollen.

  Wenn die anzuzeigenden Daten nicht von einem Datendienst stammen, ist „An Feld binden“ nur ein Textfeld. Verwenden Sie „An Feld binden“ für Spalten, die in der Datenquelle definiert sind. Beispiel: Sie geben die Namen von Spalten an, die in einer XMLList definiert sind. Wenn der angegebene Name nicht einer definierten Datenquelle entspricht, wird der Wert ignoriert und die Spalte bleibt leer.

  Spalten, die programmgesteuert zu einem DataGrid hinzugefügt werden, können nicht mit den Datenbindungsfunktionen konfiguriert werden.

- Allgemeine Eigenschaften

  Geben Sie den Kopfzeilentext und die Breite der Spalte an. Geben Sie auch an, ob die Größe der Spalte geändert werden kann und ob sie sortiert werden kann.

  Geben Sie die Breite in Pixeln an. Der Standardwert ist 100 Pixel. Wenn die `horizontalScrollPolicy`-Eigenschaft des DataGrid `false` ist, werden alle sichtbaren Spalten angezeigt. Um sicherzustellen, dass alle sichtbaren Spalten angezeigt werden, berücksichtigt das DataGrid nicht immer den angegebenen Wert für die Breite.

- Texteigenschaften

  Geben Sie Textformatierungsstile für den Text in der Spalte an.

5 Zeigen Sie die Einstellungen für alle Eigenschaften einer ausgewählten Spalte in der erweiterten Ansicht des Dialogfelds „Spalten konfigurieren“ an bzw. bearbeiten Sie sie hierüber.

Ereignisprozeduren generieren

Anwendungen, die mithilfe von Flex erstellt wurden, sind ereignisgesteuert. Komponenten auf der Benutzeroberfläche reagieren auf verschiedene Ereignisse, z. B. wenn ein Benutzer auf eine Schaltfläche klickt oder wenn die Initialisierung eines Objekts abgeschlossen ist. Sie schreiben Ereignisprozeduren in ActionScript-Code, der definiert, wie die Komponente auf das Ereignis reagiert.

Hinweis: *Sie können auch Ereignisprozeduren für nicht sichtbare Elemente, wie beispielsweise RemoteObject und HTTPService, generieren.*

Flash Builder bietet eine Ereignisprozedurhilfe, die die Ereignisprozedurfunktionen für eine Komponente generiert. Sie schreiben den Code, der das Verhalten der Komponente als Antwort auf das Ereignis definiert, in der generierten Funktion.

Sie können die Ereignisprozedurhilfe auf drei Arten aufrufen:

- Eigenschaftenansicht
- Kontextmenü für ein Element im Designmodus des MXML-Editors
- Kontexthilfe für ein Element im Quellmodus des MXML-Editors

Generierte Ereignisprozeduren

Wenn Flash Builder eine Ereignisprozedurfunktion generiert, setzt es die Ereignisprozedur in den ersten Skriptblock der Datei. Die Funktion wird an das Ende des Skriptblocks gesetzt. Die generierte Ereignisprozedur hat geschützten Zugriff und akzeptiert als einzigen Parameter die jeweilige Subklasse des Ereignisses.

Flash Builder generiert einen eindeutigen Namen für die Ereignisprozedur auf Grund des Klassennamens der Komponente oder eines benutzerdefinierten Namens der von Ihnen angegebenen Ereignisprozedur. Wenn Sie keinen benutzerdefinierten Namen angeben, wird der Name wie folgt generiert:

- Wenn eine ID-Eigenschaft definiert wird, legt Flash Builder dem Namen die ID-Eigenschaft zugrunde.
- Wenn die Komponente keine ID-Eigenschaft hat, generiert Flash Builder einen eindeutigen Namen basierend auf dem Klassennamen der Komponente.

Sie liefern den Rumpf (Body) der Ereignisprozedur. Der folgende Codeblock zeigt eine generierte Ereignisprozedur für einen Button.

```
. . .
<fx:Script>
    <![CDATA[
        protected function myButton_clickHandler(event:MouseEvent):void
        {
            // TODO Auto-generated method stub
        }

    ]]>

</fx:Script>
<s:Button label="Button" id="myButton" click="myButton_clickHandler(event)"/>
. . .
```

Flash Builder bestimmt ein Standardereignis für jede Komponente der Benutzeroberfläche. Bei einem Button ist das Standardereignis z. B. das Klickereignis. Sie können die Ereignisprozedur für das Standardereignis in der Standardansicht der Eigenschaftenansicht festlegen. Wählen Sie im Eigenschafteninspektor die Optionen „Kategorieansicht“ > „Ereignisse“, um Prozeduren für andere Ereignisse anzugeben.

Sie können auch die Inhaltshilfe in der Quellansicht verwenden, um Ereignisprozeduren zu generieren.

Ereignisprozeduren über die Eigenschaftenansicht generieren

1 Wählen Sie im Designmodus ein Objekt aus und klicken Sie dann im Eigenschafteninspektor auf „Standardansicht“.

  Im Bereich „Allgemein“ ist ein Bearbeitungsfeld für die Standardereignisprozedur zu sehen.

2 So generieren Sie eine Ereignisprozedur für das Standardereignis:

   a (Optional) Geben Sie im Textfeld „Bei *Ereignis*“ einen Namen für das Ereignis ein.

   Beispielsweise geben Sie im Textfeld „Beim Klicken“ für eine Button-Komponente **MyButtonClick** an.

   Wenn Sie keinen Namen festlegen, generiert Flash Builder einen eindeutigen Namen für das Ereignis.

   Wenn Sie einen Namen für die Ereignisprozedur angeben, können Sie auch einen Ereignisparameter festlegen. Wenn Sie keinen Ereignisparameter festlegen, generiert Flash Builder den Parameter mit einem entsprechenden Ereignistyp.

   b Klicken Sie auf das Blitzsymbol und wählen Sie „Ereignisprozedur generieren“.

   Der Editor wechselt zum Quellmodus, in der der Rumpf der generierten Ereignisprozedur hervorgehoben ist. Geben Sie die Implementierung für das Ereignis ein.

3 Um eine Ereignisprozedur für ein Ereignis für ein ausgewähltes Element zu generieren, führen Sie die folgende Schritte aus:

   a Wählen Sie die Kategorieansicht aus und erweitern Sie den Knoten „Ereignisse“, um alle Ereignisse für das Element zu sehen.

   b (Optional) Doppelklicken Sie auf den Namen des Ereignisses, um das Textfeld für den Ereignisprozedurnamen zu aktivieren. Geben Sie dann den Namen für die Ereignisprozedur ein.

   c Klicken Sie auf das Symbol im Feld „Wert“, um die Ereignisprozedur zu erstellen.

   Der Editor wechselt zum Quellmodus, in der der Rumpf der generierten Ereignisprozedur hervorgehoben ist. Geben Sie die Implementierung für das Ereignis ein.

Ereignisprozeduren über das Kontextmenü eines Elements generieren

1 Öffnen Sie in der Designansicht das Kontextmenü für ein Element.

2 Führen Sie einen der folgenden Schritte aus:

   • Wählen Sie das Standardereignis für das Element aus.

   Bei einem Button wählen Sie z. B. „Klick-Prozedur generieren“.

   • Wählen Sie „Alle Ereignisse anzeigen“ aus, um die Liste der Ereignisse in der Eigenschaftenansicht zu öffnen.

   Legen Sie über die Eigenschaftenansicht eine Ereignisprozedur fest.

Der Editor wechselt zum Quellmodus, in der der Rumpf der generierten Ereignisprozedur hervorgehoben ist. Geben Sie die Implementierung für das Ereignis ein.

Ereignisprozeduren über die Inhaltshilfe generieren

1 Erstellen Sie in der Codeansicht in einem MXML-Block eine Komponente, aber legen Sie keine Ereignisse fest.

2 Aktivieren Sie die Inhaltshilfe für die Eigenschaften einer Komponente, indem Sie nach dem Klassennamen ein Leerzeichen eingeben.

3 Wählen Sie aus der Liste der ausgewählten Eigenschaften ein Ereignis aus (z. B. „doubleClick“).

4 Drücken Sie Strg+Leertaste und wählen Sie „Ereignisprozedur generieren“.

   Flash Builder generiert einen eindeutigen Namen für die Ereignisprozedur und setzt die Ereignisprozedur in den Skriptblock.

Hinweis: *Wenn Sie einen benutzerdefinierten Namen für die Ereignisprozedur angeben, kann Flash Builder nicht die Prozedur generieren. Wenn Sie einen benutzerdefinierten Namen verwenden möchten, generieren Sie zunächst eine Ereignisprozedur und ändern Sie dann den Namen der Prozedur in der Ereigniseigenschaft und in der generierten Prozedur.*

Ereignisprozeduren für Komponenten generieren

Sie können Ereignisprozeduren für Flex-Komponenten generieren, indem Sie auf die zu den im Eigenschafteninspektor aufgelisteten Ereignissen gehörige Schaltfläche „Ereignisprozedur generieren“ klicken. Bevor Sie die Ereignisprozedur generieren, können Sie einen benutzerdefinierten Namen für die Ereignisprozedur angeben. Wenn Sie keinen benutzerdefinierten Namen festlegen, generiert Flash Builder einen Namen basierend auf der ID-Eigenschaft der Komponente. Wenn die ID-Eigenschaft nicht definiert ist, generiert Flash Builder einen eindeutigen Namen, der vom Namen der Komponente abgeleitet ist.

Ereignisprozedur für eine Komponente generieren

1. Wählen Sie in der Designansicht des Codeeditors eine Komponente aus.
2. Wählen Sie im Eigenschafteninspektor die Kategorieansicht aus, erweitern Sie die Liste der Ereignisse für das ausgewählte Element und wählen Sie das Ereignis aus, für das Sie eine Ereignisprozedur generieren möchten.
3. (Optional) Geben Sie einen Namen für die Ereignisprozedur in das Feld „Wert“ ein.
4. Klicken Sie auf die Schaltfläche „Ereignisprozedur generieren“.

   Der Editor wechselt zur Quellansicht. Flash Builder fügt die Ereignisprozedur in einen <Script>-Block ein und fügt der Komponente die Ereigniseigenschaft hinzu. Die Ereigniseigenschaft verweist auf das generierte Ereignis.

   Hinweis: *Sie können auch das Optionsmenü verwenden, um eine Ereignisprozedur für häufig verwendete Ereignisse einer Komponente zu generieren. Bei einem Button steht im Optionsmenü z. B. „Klick-Prozedur generieren“.*
5. Fügen Sie den Code für die Implementierung der Ereignisprozedur zur generierten Ereignisprozedurfunktion hinzu.

Auf Datendienste zugreifen

In Flash Builder erfolgt die Interaktion mit Daten und den datengesteuerten Elementen direkt im MXML- und ActionScript-Code. Sie können mit Daten arbeiten, automatisch Datenbankanwendungen generieren und Proxycode für Webservices sowie Code für Flex Ajax Bridge generieren und verwenden. Außerdem können Sie Sicherheitsaspekte für den Datenzugriff über den Adobe Flash Player verwalten und Flash Builder mit einem Proxydienst einsetzen.

Mit Daten in Flash Builder arbeiten

In Flash Builder werden Daten direkt im MXML- und ActionScript-Anwendungscode geändert.

Datengesteuerte Elemente und Container

stellt Steuerelement- und Container-Komponenten zur Verfügung, aus denen Sie die Benutzeroberfläche der Flex-Anwendung erstellen. Einige dieser Komponenten präsentieren Daten, die die Benutzer beim Verwenden der Anwendung auswählen und mit denen sie interagieren können. Im Folgenden finden Sie Beispiele für die Verwendung von datengesteuerten Elementen:

- Auf einem Adressformular können Benutzer ihr Land (oder andere typische Formulareingaben) über das von Ihnen auf dem Formular platzierte ComboBox- oder List-Steuerelement auswählen.
- In einer Einkaufswagen-Anwendung können Sie mit einer Spark-List-Komponente Produktdaten mit Bildern präsentieren. Sie können für die List-Komponente das Layout als VerticalLayout, HorizontalLayout oder TileLayout angeben.
- Über Container wie das TabBar- und ButtonBar-Steuerelement können Sie Standardnavigationsoptionen zur Verfügung stellen.

Daten für alle datengesteuerten Elemente werden von Ihnen über einen *Datenprovider* eingefügt.

Informationen zur Verwendung von datengesteuerten Elementen finden Sie im Abschnitt Spark list-based controls.

Datenprovider und Sammlungen

Ein *Sammlungsobjekt* enthält ein Datenobjekt (wie ein Array oder ein XMList-Objekt) und bietet eine Reihe von Methoden für den Zugriff, die Sortierung, Filterung und Änderung der Datenelemente im jeweiligen Datenobjekt. Mehrere Adobe Flex-Steuerelemente, die als *Datenprovider-Steuerelemente* bezeichnet werden, haben eine `dataProvider`-Eigenschaft, in die Sie über eine Sammlung Daten einfügen.

Das folgende einfache Beispiel zeigt, wie ein Datenprovider (als ActionScript ArrayCollection) definiert und von einem Steuerelement verwendet wird:

```
<!-- Simple example to demonstrate the Spark ComboBox control -->
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/halo">

    <fx:Script>
        <![CDATA[
            import mx.collections.ArrayCollection;

            [Bindable]
            public var complexDP:ArrayCollection = new ArrayCollection(
                [    {ingredient:"Salmon", category:"Meat"},
                    {ingredient:"Potato", category:"Starch"},
                    {ingredient:"Cucumber", category:"Vegetable"},
                    {ingredient:"Steak", category:"Meat"},
                    {ingredient:"Rice", category:"Starch"},
                    {ingredient:"Cumin", category:"Spice"}
                ]
            );

            <!-- Function to handel custom input strings -->
            private function myLabelToItemFunction(input:String):*
            {
                <!-- Returns object that matches items in dataProvider -->
                return {ingredient:input, category:"mystery"};
            }
        ]]>
    </fx:Script>
```

```
    <s:Panel title="Spark ComboBox Example" width="75%" height="75%">
        <s:layout>
            <s:VerticalLayout paddingTop="10" paddingLeft="10"/>
        </s:layout>

        <!-- Label that displayed current property values -->
        <s:Label text="Index : {cb.selectedIndex}
                       Item : {cb.selectedItem.ingredient}
                       Type : {cb.selectedItem.category}"/>

        <!-- ComboBox with custom labelToItem function -->
        <s:ComboBox
            id="cb"
            dataProvider="{complexDP}"
            width="150"
            labelToItemFunction="{myLabelToItemFunction}"
            selectedIndex="0"
            labelField="ingredient"/>
    </s:Panel>
</s:Application>
```

Weitere Informationen über Datenprovider und Sammlungen finden Sie unter Data providers and collections.

Remotedatenzugriff

Flex enthält Datenzugriffskomponenten, die auf einer serviceorientierten Architektur (SOA) basieren. Diese Komponenten verwenden Remoteprozeduraufrufe, um mit Serverumgebungen wie z. B. PHP, Adobe ColdFusion® und Microsoft ASP.NET zu interagieren, um Flex-Anwendungen mit Daten zu versorgen und um Daten an Backend-Datenquellen zu senden.

Je nach den für eine bestimmte serverseitige Anwendung vorhandenen Schnittstellentypen können Sie auf eine der folgenden Methoden eine Verbindung zu einer Anwendung herstellen:

- HTTP GET oder POST über die HTTPService-Komponente
- SOAP-kompatible Webservices über die WebService-Komponente
- Remoting-Dienste in Adobe AMF (Action Message Format) über die RemoteObject-Komponente

Hinweis: *Wenn Sie mit Flash Builder Anwendungen entwickeln, die auf serverseitige Daten zugreifen, und Sie auf diese Daten über eine andere Domäne als die Domäne zugreifen, von der die Anwendung geladen wurde, verwenden Sie eine* `cross-domain.xml`*-Datei oder einen Proxy. Siehe „Flash Player-Sicherheit verwalten“ auf Seite 254.*

Mit Flash Builder können Sie auch Anwendungen erstellen, die Adobe ADEP Data Services (ein separates Produkt, das erweiterte Datendienstfunktionen bietet) verwenden. ADEP Data Services bietet Proxyfunktionen für RPC-Dienstanwendungen (Remote Procedure Call, Remoteprozeduraufruf) sowie erweiterte Sicherheitskonfiguration. ADEP Data Services stellt auch die folgenden Datendienste zur Verfügung:

Datenverwaltungsdienst Ermöglicht die Erstellung von Anwendungen, die verteilte Daten nutzen. Der Datenverwaltungsdienst ermöglicht außerdem die Verwaltung großer Mengen von Daten und verschachtelter Datenbeziehungen wie z. B. One-to-One und One-to-Many.

Messaging-Dienst Ermöglicht die Erstellung von Anwendungen, die Meldungen an andere Anwendungen senden und von anderen Anwendungen empfangen können, darunter Anwendungen, die in Flex erstellt wurden, und JMS-Anwendungen (Java Message Service).

Flash Builder bietet Assistenten und Werkzeuge, die mit Datendiensten verbunden werden und Datendienstvorgänge an Anwendungskomponenten binden. Weitere Informationen finden Sie unter Erstellen datenorientierter Anwendungen mit Flash Builder.

Datenbindung

Im Codebeispiel unter „Datenprovider und Sammlungen“ auf Seite 222 haben Sie vielleicht schon gesehen, dass der Wert der Eigenschaft `dataProvider` des ComboBox-Steuerelements `"{complexDP}"` lautet. Dies ist ein Beispiel für eine Datenbindung.

Bei der Datenbindung wird der Wert eines Objekts (die Quelle) in ein anderes Objekt (das Ziel) kopiert. Wenn ein Objekt an ein anderes Objekt gebunden ist, werden die an der Quelle vorgenommenen Änderungen automatisch im Ziel übernommen.

Im folgenden Beispiel wird die Texteigenschaft eines TextInput-Steuerelements (die Quelle) mit der Texteigenschaft eines Label-Steuerelements (dem Ziel) verbunden. Dadurch wird der in das TextInput-Steuerelement eingegebene Text im Label-Steuerelement angezeigt:

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/mx" minWidth="955" minHeight="600">
    <fx:Declarations>
        <!-- Place non-visual elements (e.g., services, value objects) here -->
    </fx:Declarations>

    <s:TextInput id="LNameInput"  x="10" y="10"/>
    <s:Label text="{LNameInput.text}"  x="10" y="50"/>
</s:Application>
```

Um Daten von einem Objekt mit einem anderen zu verbinden, verwenden Sie entweder die Syntax aus geschweiften Klammern ({ }, siehe Beispiel) oder das `<fx:Binding>`-Tag. Weitere Informationen hierzu finden Sie unter Verwenden von Datenbindung bei Datenmodellen und Datenbindung.

Datenmodelle

Ein *Datenmodell* ist ein Objekt, mit dem Sie Daten vorübergehend speichern können, um sie leicht bearbeiten zu können. Sie können ein Datenmodell in ActionScript oder MXML definieren, indem Sie ein Tag (wie z. B. `<fx:Model>` oder ein beliebiges Objekt, das Eigenschaften enthält, verwenden. Im Beispiel des folgenden Datenmodells werden Informationen wie der Name, Alter und Telefonnummer einer Person angezeigt:

```
<fx:Declarations>
    <fx:Model id="reg">
        <registration>
            <name>{nme.text}</name>
            <email>{email.text}</email>
            <phone>{phone.text}</phone>
            <zip>{zip.text}</zip>
            <ssn>{ssn.text}</ssn>
        </registration>
    </fx:Model>
</fx:Declarations>
```

Die Felder in einem Datenmodell können entweder statische Daten (wie im Beispiel) enthalten, oder Sie können per Datenbindung Daten an ein Datenmodell übergeben und von dort beziehen.

Außerdem können Sie das Datenmodell in einer XML-Datei definieren. Anschließend referenzieren Sie die XML-Datei über das Dateisystem oder über eine URL, indem Sie vom `<fx:Model>`-Tag die Eigenschaft `source` verwenden (siehe folgendes Beispiel):

```
<fx:Model source="content.xml" id="Contacts"/>
<fx:Model source="http://www.somesite.com/companyinfo.xml" id="myCompany"/>
```

Weitere Informationen zu Datenmodellen finden Sie unter Speichern von Daten.

Daten validieren

Die Validierung von Daten dient zur Überprüfung, ob die vom Benutzer in die Anwendung eingegebenen Daten gültig sind. Wenn der Benutzer z. B. eine gültige Postleitzahl eingeben soll, können Sie einen *Datenvalidierer* für Postleitzahlen verwenden.

Flex enthält vordefinierte Datenvalidierer für die folgenden Datentypen: Kreditkarte, Währung, Datum, E-Mail, Nummer, Telefonnummer, regulärer Ausdruck, Sozialversicherungsnummer, String und Postleitzahl.

Datenvalidierer sind keine visuellen Flex-Komponenten. Sie greifen auf sie also nicht vom Komponentenbedienfeld aus zu, sondern arbeiten mit ihnen im Code (wie im folgenden MXML-Beispiel gezeigt):

```
<!-- Define the ZipCodeValidator. -->
<mx:ZipCodeValidator id="zcV" source="{zipcodeInput}" property="text"/>
<!-- Define the TextInput control for entering the zip code. -->
<s:TextInput id="zipcodeInput"/>
```

In diesem MXML-Beispiel wird der Validierer durch das entsprechende MXML-Tag definiert und ist an die ID-Eigenschaft eines TextInput-Steuerelements gebunden. Wenn der Benutzer zur Laufzeit die Telefonnummer in das TextInput-Steuerelement eingibt, wird die Nummer sofort geprüft.

Sie können Datenvalidierer in ActionScript verwenden, indem Sie eine Variable als Instanz einer Validiererklasse definieren und dann eine Funktion zum Binden der Variable an das Eingabesteuerelement erstellen.

Datenvalidierer werden häufig bei Datenmodellen verwendet. Weitere Informationen hierzu finden Sie unter Validating Data.

Daten formatieren

Um das richtige Format für bestimmte Datentypen in der Anwendung anzuzeigen, können Sie einen *Datenformatierer* verwenden. Flex enthält vordefinierte Datenformatierer für die folgenden Datentypen: Währung, Datum, Nummer, Telefon und Postleitzahl.

Datenformatierer werden an Eingabesteuerelemente gebunden und formatieren Daten, die vom Benutzer eingegeben wurden, richtig. Beispielsweise kann ein Benutzer ein Datum im folgenden Format eingeben:

120105

An das Texteingabesteuerelement ist ein Datenformatierer gebunden, die das Datum im folgendem Format speichert und anzeigt:

12/01/05

Genau wie Datenvalidierer sind auch Datenformatierer keine visuellen Flex-Komponenten, mit denen Sie als MXML-Tags oder ActionScript-Klassen arbeiten können.

Weitere Informationen hierzu finden Sie unter Formatting Data.

Zugriff auf Datendienste konfigurieren

Der Projekterstellungsassistent für Flex-Projekte und Flex-Mobilprojekte stellt Optionen zum Konfigurieren des Zugriffs auf Datendienste bereit. Die Schritte im Assistenten sind für Flex-Projekte und Flex-Mobilprojekte dieselben. Sie können Zugriff auf folgende Dienstarten konfigurieren:

PHP-Dienste
ColdFusion-Dienste
Java-Dienste
ASP.NET

Zugriff auf PHP-Dienste konfigurieren

Damit auf Daten von PHP-Diensten zugegriffen werden kann, muss ein Server, der die Dienste hostet, verfügbar sein. Der Server kann entweder ein lokaler Server oder ein Server aus einem lokalen Netzwerk sein.

1 Geben Sie im Projekterstellungsassistenten den Projektpfad an. Geben Sie für Mobilprojekte die Mobileinstellungen an.

Siehe „Flex-Projekte“ auf Seite 90 und „Flex-Mobilprojekte“ auf Seite 35.

2 Wählen Sie als Anwendungsservertyp „PHP“.

3 Geben Sie Webstamm und Stamm-URL für den Dienst an. Klicken Sie auf „Konfiguration validieren“.

Normalerweise geben Sie einen Webstamm und eine Stamm-URL an, der bzw. die in Ihrer Umgebung lokal vorhanden sind. Sie können jedoch auch auf Netzwerkserver zugreifen. Stellen Sie sicher, dass der Ordner freigegeben ist und dass das Konto, auf dem Flash Builder ausgeführt wird, Schreibzugriff gewährt.

Stellen Sie sicher, dass ein Laufwerk für den Netzwerkserver zugeordnet oder angeschlossen ist. Geben Sie dann einen Pfad zum Server an. Der Pfad ist plattformspezifisch. Beispiel:

(Windows) `\\10.192.18.12\Server\Webstamm`

(Windows) `Z:\Webstamm`

(Mac) `/Volumes/Webstamm`

4 (Optional) Legen Sie den Ausgabeordner für die Anwendung fest.

5 Klicken Sie auf „Fertig stellen“ oder auf „Weiter“, um weitere Konfigurationsoptionen auszuwählen.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

Zugriff auf ColdFusion-Dienste konfigurieren

Um auf Daten zuzugreifen, die ColdFusion verwenden, müssen Sie Adobe ColdFusion® 8 oder Adobe ColdFusion 9 haben. Weitere Informationen hierzu finden Sie auf der Produktseite für ColdFusion.

1 Geben Sie im Projekterstellungsassistenten den Projektpfad an. Geben Sie für Mobilprojekte die Mobileinstellungen an.

Siehe „Flex-Projekte“ auf Seite 90 und „Flex-Mobilprojekte“ auf Seite 35.

2 Wählen Sie als Anwendungsservertyp „ColdFusion“ aus und aktivieren Sie dann die gewünschten Optionen:

Remote Object Access Service verwenden Wenn Sie die Option „Remote Object Access Service verwenden“ deaktivieren, geben Sie im nächsten Schritt den Webstamm und die Webstamm-URL an.

Wenn Sie die Option „Remote Object Access Service verwenden“ aktivieren, stehen die folgenden Optionen zur Wahl:

- ADEP Data Services

  Legen Sie ADEP Data Services nur als einen ColdFusion-Anwendungstyp fest, wenn Ihre ColdFusion 8-Installation für ADEP Data Services 2.6.1 (früher LiveCycle Data Services genannt) konfiguriert ist. Siehe Integrieren von Adobe LiveCycle Data Services ES 2.6 in Adobe ColdFusion 8.

  Normalerweise geben Sie für ADEP Data Services als Anwendungsservertyp „Java“ an, nicht „ColdFusion“. Siehe „Zugriff auf Java-Dienste konfigurieren“ auf Seite 228.

- BlazeDS

  Geben Sie „BlazeDS“ nur dann als ColdFusion-Anwendungstyp an, wenn Ihre ColdFusion 8-Installation für Adobe BlazeDS 3.1 konfiguriert ist. Siehe Integrating BlazeDS with a ColdFusion 8 Installation.

  Normalerweise geben Sie für „BlazeDS“ als Anwendungsservertyp „Java“ an, nicht „ColdFusion“. Siehe „Zugriff auf Java-Dienste konfigurieren“ auf Seite 228.

- ColdFusion Flash Remoting

  Verwenden Sie diese Option, wenn Sie planen, datenorientierte Entwicklungswerkzeuge zu verwenden, die in Flash Builder verfügbar sind. Diese Option gilt auch, wenn Sie Flash Remoting verwenden, um in ColdFusion-Komponenten Methoden aufzurufen. Weitere Informationen finden Sie unter Erstellen datenorientierter Anwendungen mit Flash Builder.

3 Geben Sie eine Serveradresse, einen Webstamm, eine Webstamm-URL und einen Kontextstamm an:

Beim Zugriff auf einen Remoteobjektdienst können Sie eine eigenständige ColdFusion-Konfiguration oder eine ColdFusion-Konfiguration einrichten, die auf einem Java-Anwendungsserver bereitgestellt wird:

- Eigenständig

  Verwenden Sie die Option „Eigenständig“, wenn Ihre ColdFusion-Installation die Serverkonfiguration verwendet.

  Geben Sie den Speicherort des ColdFusion-Servers, den Speicherort des Webstamms und die Webstamm-URL an.

- Auf Java-Anwendungsserver bereitgestellt

  Verwenden Sie die Option „Auf Java-Anwendungsserver bereitgestellt“, wenn Ihre ColdFusion-Installation die Konfiguration mit mehreren Servern oder die Java-Konfiguration verwendet.

  Geben Sie einen Webstamm, eine Stamm-URL und einen Kontextstamm an: Wenn Sie die ColdFusion-Konfiguration mit mehreren Servern verwenden, brauchen Sie den Kontextstamm normalerweise nicht anzugeben.

  Der Kontextstamm stimmt normalerweise mit dem letzten Segment des URL-Stammordners überein, wenn Sie ColdFusion als Webanwendung in der ColdFusion-Java-Konfiguration bereitstellen.

Wenn Sie den Speicherort des Server- und Webstamms angeben, navigieren Sie zu einem lokalen Ordner oder geben Sie einen Pfad zu einem Ordner auf einem Netzwerkserver an. Stellen Sie sicher, dass der Ordner freigegeben ist und dass das Konto, auf dem Flash Builder ausgeführt wird, Schreibzugriff gewährt.

Stellen Sie sicher, dass ein Netzlaufwerk für den Netzwerkserver zugeordnet oder angeschlossen ist. Der Pfad zu einem Netzwerkserver ist plattformspezifisch. Beispiel:

(Windows) `\\10.192.18.12\Server\Webstamm`

(Windows) `Z:\Webstamm`

(Mac) `/Volumes/Webstamm`

4 Klicken Sie auf „Konfiguration validieren“, um sicherzustellen, dass die Einrichtung korrekt ist.

Wenn das Schreiben im Webstammordner nicht möglich ist, zeigt Flash Builder eine Warnung an.

5 Wählen Sie einen Ausgabeordner für die kompilierte Anwendung.

6 Klicken Sie auf „Fertig stellen“ oder auf „Weiter“, um weitere Konfigurationsoptionen auszuwählen.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

Zugriff auf Java-Dienste konfigurieren

Mit dieser Projektkonfiguration können Sie Flex-Projekte erstellen, die eine auf Java basierende Service-Klasse mit der Option „Remote Object Access Service“ verwenden. Wenn keine Option ausgewählt ist und ein Java-Server verwendet wird, wird ein Ausgabeordner unter dem Serverstamm erstellt. Wenn Sie das Eclipse-Plug-In „Web Tools Project (WTP)“ installiert haben, können Sie kombinierte Java/Flex-Projekte mit oder ohne Remote Object Access Service erstellen.

Hinweis: *ADEP Data Services (früher LiveCycle Data Services genannt) und BlazeDS unterstützen bestimmte Versionen des Flex SDK. Überprüfen Sie die ADEP Data Services-Kompatibilitätsmatrix, um zu ermitteln, welche Versionen des Flex SDK Ihre Version von ADEP DataService unterstützt. In der Kompatibilitätsmatrix werden außerdem die von BlazeDS unterstützten Versionen des Flex SDK aufgeführt.*

1 Geben Sie im Projekterstellungsassistenten den Projektpfad an. Geben Sie für Mobilprojekte die Mobileinstellungen an.

Siehe „Flex-Projekte“ auf Seite 90 und „Flex-Mobilprojekte“ auf Seite 35.

2 Wählen Sie als Anwendungsservertyp „Java“.

3 (Optional) Aktivieren Sie die Option „Remote Object Access Service verwenden“.

Data Services ES wird automatisch aktiviert. Sie können „BlazeDS“ wählen. Wenn Sie WTP installiert haben, können Sie auch ein kombiniertes Java/Flex-Projekt erstellen, das WTP verwendet (der Java-Quellordner wird für Sie ausgewählt).

4 Konfigurieren Sie den Java-Anwendungsserver.

- Wenn Sie die Optionen „Remote Object Access Service verwenden“ und „Data Services ES“ oder „BlazeDS“ ausgewählt haben, legen Sie die Stammeinstellungen fest:

  Stammordner Speicherort des Webanwendungsservers, der Ihre Anwendungsdaten bereitstellt (Beispiel: `C:\Programme\Apache Software Foundation\Tomcat 5.5\webapps\myservices`). Wenn Sie einen Remoteserver verwenden, muss er sich auf einem zugeordneten Laufwerk befinden oder über UNC zugreifbar sein.

  Stamm-URL Stamm-URL der Webanwendung. Diese URL entspricht dem Speicherort, der im obigen Stammordner genannt wird. Für BlazeDS auf Tomcat kann diese URL folgendermaßen lauten:

  ```
  http://localhost:8080/myservices
  ```

  Für Data Services lautet die standardmäßige Stamm-URL wie folgt:

  ```
  http://localhost:8400/lcds/
  ```

  Wenn Sie einen Remoteserver verwenden, kann die URL folgendermaßen lauten:

  ```
  http://myserver.com:8400/lcds/.
  ```

  Kontextstamm Der Kontextstamm stimmt normalerweise mit dem letzten Segment des URL-Stammpfades überein. Für die obigen Beispiele in der Stamm-URL lautet der Kontextstamm `/myservices` für BlazeDS und `/lcds` für Data Services.

- Wenn Sie die Option „Kombiniertes Java/Flex-Projekt mit WTP erstellen“ (mit oder ohne ADEP Data Services) aktiviert haben:
  - Geben Sie die Namen der Java- und Flex-Quellordner und der Ziel-Laufzeitumgebung an.

    Wenn Sie ein Flex-Projekt mit ADEP Data Services erstellen, erstellt Flash Builder entweder einen Ordner mit demselben Namen wie das Projekt oder verwendet einen vorhandenen Ordner mit diesem Namen. Der Ordner ist ein Unterordner des Stammordners, den Sie für das Projekt festgelegt haben.
  - Legen Sie bei Data Services ES eine flex.war-Datei fest, die sich im Serverinstallationsordner befindet.

Hinweis: *Unabhängig von der von Ihnen gewählten Option für ein ADEP Data Services-Projekt in Flash Builder müssen Sie einen gültigen Stammordner und eine Stamm-URL angeben. Diese Werte legen den Stamm einer ADEP Data Services-Webanwendung fest. Wenn Sie die Optionen deaktivieren, müssen Sie nur den Webstamm und die Stamm-URL eingeben.*

5 Geben Sie den Pfad an, in dem das Projekt kompiliert werden soll.

6 Klicken Sie auf „Fertig stellen“ oder auf „Weiter“, um weitere Konfigurationsoptionen auszuwählen.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

Flex-Projekte mit ASP.NET erstellen

Wenn Microsoft Windows und Microsoft Visual Web Developer installiert sind, können Sie Flex-Projekte erstellen, die ASP.NET für die Bereitstellung verwenden. Wenn Sie außerdem Zugang zu einem Internet Information Services (IIS)-Entwicklungsserver haben, können Sie Flex-Projekte mit einem Flex-Ausgabeordner unter IIS erstellen.

1 Geben Sie im Projekterstellungsassistenten den Projektpfad an. Geben Sie für Mobilprojekte die Mobileinstellungen an.

Siehe „Flex-Projekte“ auf Seite 90 und „Flex-Mobilprojekte“ auf Seite 35.

2 Wählen Sie als Anwendungsservertyp „ASP.NET“.

3 Wählen Sie den ASP.NET-Server.

- Wenn Sie einen ASP.NET Development Server verwenden, ist es nicht notwendig, einen Serverpfad anzugeben.
- Wenn Sie IIS verwenden, geben Sie den Webanwendungsstamm und die Webanwendungs-URL ein.
- Legen Sie den Ausgabeordner für die Anwendung fest.

4 Klicken Sie auf „Fertig stellen“ oder auf „Weiter“, um weitere Konfigurationsoptionen auszuwählen.

Siehe „Erstellungspfade, native Erweiterungen und sonstige Projektkonfigurationsoptionen“ auf Seite 96.

Serveroptionen für vorhandene Projekte ändern

In bestimmten Situationen erfüllt die Originalserverkonfiguration für ein Projekt u. U. nicht Ihre Anforderungen. In diesen Fällen können Sie die Serverkonfiguration für eine Web- oder Desktopanwendung über das Fenster „Projekteigenschaften“ neu konfigurieren.

Wählen Sie im Fenster „Projekteigenschaften“ die Option „Flex Server“ aus, um Serveroptionen für das Projekt hinzuzufügen oder zu ändern:

- Wählen Sie „Keiner“ aus, um die Serverkonfiguration von einem Projekt zu entfernen.

  Wenn Sie die Serverkonfiguration von einem Projekt entfernen, werden hierdurch alle hinzugefügten SWC-Dateien auf dem Bibliothekspfad für den jeweiligen Servertyp entfernt.

- Wählen Sie einen Servertyp aus, der geändert werden soll, oder fügen Sie die Serverkonfiguration eines Projekts hinzu.

  Alle Serveroptionen für die ausgewählte Serverkonfiguration stehen zur Verfügung. Weitere Informationen zu Serverkonfigurationseinstellungen finden Sie im Abschnitt „Flex-Projekte“ auf Seite 90.

  Wenn Sie den Servertyp eines Projekts ändern, können hierdurch Fehler im bestehenden Code, der für den ursprünglichen Servertyp geschrieben wurde, verursacht werden. Analysieren und korrigieren Sie etwaige Fehler im Code.

Anwendungen überwachen, die auf Datendienste zugreifen

Die Netzwerküberwachung ist ein nützliches Werkzeug zur Überwachung und zum Debuggen von Anwendungen, die auf Datendienste zugreifen. Mit der Netzwerküberwachung können Sie Daten überprüfen, die zwischen einer Anwendung und einem Datendienst ausgetauscht werden. Außerdem werden XML-, AMF- und JSON-Daten geprüft, die mithilfe von SOAP-, AMF-, HTTP- und HTTPS-Protokollen gesendet werden.

Die Netzwerküberwachung ist in der Flash-Entwicklungsperspektive und der Flash-Debugperspektive aktiv.

Netzwerküberwachung aktivieren

Die Netzwerküberwachung wird für einzelne Flex-Projekte aktiviert. Der Überwachungsstatus („aktiviert“ oder „deaktiviert“) bezieht sich auf alle Anwendungen im jeweiligen Projekt. Es ist nicht möglich, die Netzwerküberwachung für eine einzelne Anwendung zu aktivieren bzw. zu deaktivieren.

Die Netzwerküberwachung ist standardmäßig nicht aktiviert. Sie können die Netzwerküberwachung aktivieren, indem Sie in der Symbolleiste der Ansicht „Netzwerküberwachung“ auf das Symbol für „Überwachung aktivieren“ klicken.

Diese Schritte setzen voraus, dass Sie die Flex-Entwicklungsperspektive oder die Flex-Debugperspektive aktiviert haben.

1. Wenn die Netzwerküberwachungsansicht nicht geöffnet ist, wählen Sie im Flash Builder-Menü „Fenster“ > „Ansicht anzeigen“ > „Andere“ > „Flash Builder“ > „Netzwerküberwachung“.
2. Wenn die Netzwerküberwachung nicht aktiviert ist, klicken Sie in der Symbolleiste der Netzwerküberwachung auf das Symbol „Netzwerküberwachung aktivieren“. Mit diesem Symbol aktivieren bzw. deaktivieren Sie die Netzwerküberwachung.

Remotedienste überwachen

Um die Anwendung zu überwachen, führen Sie entweder die Entwicklungs- oder die Debugversion der Anwendung mit aktivierter Netzwerküberwachung aus.

Die Netzwerküberwachung erfasst und speichert generell alle Ereignisdaten, bis Sie die Anwendung entweder beenden oder explizit die Daten löschen. Die Ereignisse werden in chronologischer Reihenfolge angezeigt.

Überwachungssitzungen starten

1. Führen Sie eine Entwicklungs- oder Debugversion der Anwendung aus, die auf Remotedienste zugreift.

2 Für jeden Zugriff auf einen Remotedienst listet die Netzwerküberwachung folgende Informationen auf:

- Uhrzeit der Anforderung;
- Anfordernder Dienst;
- Methode und URL, sofern zutreffend;
- Uhrzeit der Antwort;
- Verstrichene Zeit.

3 Wählen Sie eine Spaltenüberschrift aus, um die zurückgegebenen Daten entsprechend den Werten in der ausgewählten Spalte zu sortieren.

Klicken Sie erneut auf die Spalte, um die Reihenfolge der Daten umzukehren.

4 Wählen Sie die Registerkarte „Request" und „Parameter" im unteren Bereich der Überwachung aus, um die Details zur Request-Methode anzuzeigen.

Über diese Registerkarten können die tatsächlichen im Request gesendeten Daten sowie andere Informationen zum Request eingesehen werden.

5 Wählen Sie die Registerkarte „Antwort" und „Ergebnis" im unteren Bereich der Überwachung aus, um die Details zur Antwort anzuzeigen.

Über diese Registerkarten können die tatsächlichen in der Antwort gesendeten Daten sowie andere Informationen zur Antwort eingesehen werden.

6 Doppelklicken Sie auf einen Eintrag, um zum Quellcode der jeweiligen Methode zu gelangen.

Der Flash Builder-Quelleditor wird geöffnet und die relevante Quellcodezeile ist hervorgehoben.

Hinweis: *Bei den meisten Ereignissen kann die Netzwerküberwachung ein Ereignis mit dem Flex-Quellcode korrelieren. Bei einigen Ereignissen, die außerhalb des Bereichs der Netzwerküberwachung ausgelöst werden, kann die Überwachung den Flex-Quellcode nicht finden.*

7 Klicken Sie in der Symbolleiste der Netzwerküberwachung auf das Symbol „Speichern", um alle bei der Überwachung erfassten Informationen als XML-Datei zu speichern.

Hinweis: *Verwenden Sie die generierte XML-Datei, um die Daten offline zu analysieren. Die Daten aus dieser Datei können nicht wieder in die Netzwerküberwachung importiert werden.*

8 Klicken Sie in der Symbolleiste der Netzwerküberwachung auf das Symbol „Löschen", um alle aus der Überwachung erfassten Informationen zu entfernen.

Überwachungssitzungen unterbrechen

Die Netzwerküberwachung kann unterbrochen und wieder fortgesetzt werden. Wenn Sie eine Sitzung unterbrechen und dann wieder fortsetzen, sind hiervon alle Anwendungen im Flex-Projekt betroffen. Sie können also z. B. nicht eine Anwendung im Projekt unterbrechen und eine andere Anwendung weiterhin überwachen.

1 Klicken Sie in der Symbolleiste der Netzwerküberwachung auf die Schaltfläche „Unterbrechen", um die Überwachung einer Sitzung zu unterbrechen.

2 Klicken Sie in der Symbolleiste auf die Schaltfläche „Fortsetzen", um die Sitzung weiter zu überwachen.

Überwachungssitzungen anhalten

Um eine Überwachungssitzung anzuhalten, deaktivieren Sie die Netzwerküberwachung.

1 (Optional) Schließen Sie die Netzwerküberwachung.

Hinweis: *Wenn Sie nur die Netzwerküberwachung schließen, wird die Überwachungssitzung hierdurch nicht angehalten. Die Überwachung ist weiterhin aktiv, selbst wenn die Netzwerküberwachung geschlossen ist.*

2 Klicken Sie auf die Schaltfläche „Überwachung aktivieren".

Mit diesem Symbol aktivieren bzw. deaktivieren Sie die Netzwerküberwachung.

Hinweis: *Wenn Sie die Netzwerküberwachung deaktivieren, ist sie für alle Anwendungen im Flex-Projekt deaktiviert.*

Unterstützung des HTTPS-Protokolls

Die Netzwerküberwachung unterstützt die Überwachung von HTTPS-Aufrufen von Servern, die von einer Zertifizierungsstelle zertifiziert sind oder über ein selbstsigniertes Zertifikat verfügen.

Um Aufrufe über das HTTPS-Protokoll zu überwachen, können Sie die Standardvoreinstellung für die Netzwerküberwachung ändern, sodass SSL-Sicherheitsprüfungen ignoriert werden. Öffnen Sie das Dialogfeld „Voreinstellungen" und wählen Sie die Optionen „Flash Builder" > „Netzwerküberwachung".

Netzwerküberwachungsdaten anzeigen

Im Bedienfeld am linken Rand der Ansicht „Netzwerküberwachung" werden Informationen über die Quelle der Daten angezeigt. Die folgenden Informationen werden angezeigt:

- Quell-URL für den Datendienst
- Typ des angezeigten Dienstes

  Z. B. RemoteService, HTTPService oder WebService
- Die Anforderungszeit, Antwortzeit und verstrichene Zeit für die Datenanforderung
- Der Name des Vorgangs, der vom Datendienst aufgerufen wurde

Die Ansicht „Netzwerküberwachung" verfügt zur Datenanzeige über die beiden Registerkarten „Anforderung" und „Antwort", auf denen Sie die Anforderungs- und Antwortdaten ablesen können.

Unter „Anforderung" und „Antwort" können Sie Daten jeweils in der Baum-, Quell- und Hex-Ansicht anzeigen. Klicken Sie auf das gewünschte Ansichtssymbol, um die Datenanzeige der Netzwerküberwachung zu ändern.

- Baumansicht

  Zeigt die XML-, JSON- und AMF-Daten im Baumstrukturformat an. Dies ist die Standardeinstellung für Daten.
- Quellansicht

  Zeigt die übertragenen Rohdaten an.
- Hex-Ansicht

  Zeigt die Daten in Hexadezimalformat an. Die Hex-Ansicht ist nützlich, wenn Sie Binärdaten debuggen, die über ein Netzwerk übertragen wurden.

Standardmäßig löscht die Netzwerküberwachung bei jedem Start einer Anwendung alle erfassten Daten. Sie können jedoch das Standardverhalten ändern und sämtliche Überwachungsdaten beibehalten. Zu den beibehaltenen Daten gehören die Daten von allen Anwendungen in allen Projekten. Öffnen Sie das Dialogfeld „Voreinstellungen" und wählen Sie die Optionen „Flash Builder" > „Netzwerküberwachung". Deaktivieren Sie die Option „Einträge beim Start löschen".

Netzwerküberwachungsdaten speichern

Sie können Netzwerküberwachungsdaten als XML-Datei speichern. Klicken Sie in der Symbolleiste der Ansicht „Netzwerküberwachung" auf die Schaltfläche „Speichern", um die Netzwerküberwachungsdaten zu speichern.

Mehrere Anwendungen überwachen

Sie können mehrere Anwendungen gleichzeitig überwachen. Für die Überwachung mehrerer Anwendungen gibt es zwei Szenarien:

- Überwachen mehrerer Anwendungen in einem Projekt

  Sie können nur eine Netzwerküberwachung pro Flex-Projekt haben. Wenn Sie mehrere Anwendungen in einem Projekt überwachen, werden in der Überwachung Ereignisse von allen Anwendungen entsprechend dem Zeitpunkt des Auftretens des jeweiligen Ereignisses angezeigt.

  Es ist nicht möglich, Ereignisse in der Überwachung nach bestimmten Anwendungen zu filtern.

- Überwachen mehrerer Anwendungen in verschiedenen Projekten

  Sie können eine Netzwerküberwachung für jedes aktive Flex-Projekt öffnen. Jede Netzwerküberwachung ist unabhängig und zeigt nur die Ereignisse für ihr jeweiliges Projekt an.

  Wenn Sie eine Netzwerküberwachung in einem Projekt unterbrechen oder deaktivieren, unterbrechen bzw. deaktivieren Sie hierdurch nicht die Netzwerküberwachung in anderen Projekten.

Mobilanwendungen überwachen

Mit der Netzwerküberwachung können Sie Daten überprüfen, die zwischen einer Mobilanwendung und einem Datendienst ausgetauscht werden.

Sie können die Netzwerküberwachung aktivieren, sodass sie Mobilanwendungen überwacht, indem Sie folgende Schritte durchführen:

1 Führen Sie die Debugkonfiguration der Anwendung aus, während die Netzwerküberwachung aktiviert ist.

2 Geben Sie eine Startmethode an.

   - **Auf Desktop** Wählen Sie diese Startmethode, wenn Sie nicht über ein Mobilgerät verfügen und Sie die Anwendung auf dem Desktop überwachen möchten.

     Wählen Sie „Auf AIR-Simulator“, um die Anwendung auf einem simulierten Gerät zu überwachen, das unter Verwendung von AIR Debug Launcher (ADL) erstellt wurde. Das Gerät wird entsprechend der von Ihnen gewählten Gerätekonfiguration simuliert.

   - **Auf Gerät** Wählen Sie diese Startmethode zum Überwachen der Anwendung auf einem Mobilgerät.

     Normalerweise verwenden Sie die Startmethode „Auf Gerät“, um Anwendungen, die auf den nativen Code des Geräts zugreifen, zu überwachen. Flash Builder kann auf das Gerät entweder über den USB-Port Ihres Computers oder über das Netzwerk über Wi-Fi zugreifen.

     Bei beiden Optionen zeichnet die Netzwerküberwachung Aufrufe nur auf, wenn das Gerät und der Hostcomputer an dasselbe Netzwerk über Wi-Fi angeschlossen sind.

     Wenn die Netzwerküberwachung keine Verbindung über Wi-Fi herstellen kann, wird in Flash Builder ein Dialogfeld angezeigt, das die IP-Adresse des Hostcomputers anfordert. Nachdem die Verbindung hergestellt ist, werden die Netzwerkaufrufe von der Anwendung über das Wi-Fi-Netzwerk des Geräts an Flash Builder umgeleitet.

3 Führen Sie die Überwachung für die Mobilanwendung wie für eine Web- oder Desktopanwendung durch, die auf Datendienste zugreift. Weitere Informtionen finden Sie unter Anwendungen überwachen, die auf Datendienste zugreifen.

Beschränkungen der Netzwerküberwachung

Beachten Sie beim Überwachen von Netzwerkdaten die folgenden Beschränkungen:

- Die Netzwerküberwachung unterstützt keine Anwendungen, die mithilfe von reinen ActionScript- und Bibliotheksprojekten erstellt wurden.
- Die Netzwerküberwachung bietet keine Unterstützung für das Real Time Messaging Protocol (RTMP). Beispielsweise ist es nicht möglich, Streaming-Video zu überwachen.

Flex-Bibliotheksprojekte

Mit Bibliotheksprojekten können Sie benutzerdefinierte Codebibliotheken erstellen, die Sie in anderen Anwendungen nutzen bzw. an andere Entwickler weitergeben können. Ein Bibliotheksprojekt generiert eine SWC-Datei, die eine Archivdatei für Flex-Komponenten und andere Ressourcen ist. Das Flex-Framework ist beispielsweise in SWC-Dateien enthalten. Wenn Sie ein Flex-Projekt erstellen, werden die SWC-Dateien des Flex-Frameworks zum Bibliothekspfad des Projekts hinzugefügt. Sie können den Bibliothekspfad anzeigen und bearbeiten, indem Sie die Eigenschaftenseite mit dem Erstellungspfad für das Projekt aufrufen (wählen Sie für Flex-Projekte „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“).

In einer SWC-Datei ist eine SWF-Datei archiviert, die Komponenten, Ressourcen und eine Datei mit dem Namen „catalog.xml“ (das Manifest der in der SWF-Datei enthaltenen Elemente) enthält. Die SWF-Datei enthält typischerweise eine oder mehrere Komponenten sowie alle anderen erforderlichen Ressourcen. Wenn Sie die Bibliothek zu einem Projekt hinzufügen, können Sie diese Komponenten in der Anwendung verwenden und für diese Komponenten stehen dann auch Codehinweise zur Verfügung.

Darüber hinaus bieten SWC-Bibliotheken nicht nur eine praktische Methode, Komponenten zu verpacken und zu verteilen, sondern sie werden auch als Designs (das visuelle Aussehen von Flex-Anwendungen) verwendet. Eine SWC-Designdatei besteht aus einer CSS-Datei und allen zugehörigen Grafikelementen. Weitere Informationen zum Erstellen und Verwenden von Designs finden Sie unter About themes.

Bibliotheken sind praktisch, wenn Sie Komponenten komplett in ActionScript erstellen und sie in Flash Builder im Designmodus verwenden. ActionScript-Komponenten werden im Designmodus nicht visuell wiedergegeben; sie müssen zuerst in einer SWF-Datei kompiliert werden. Wenn Sie ActionScript-Komponenten zu einem Bibliotheksprojekt hinzufügen, erstellen Sie eine SWF-Datei, die in einer SWC-Datei enthalten ist. Sie können die Bibliothek zu einem Bibliothekspfad des Projekts hinzufügen. Die ActionScript-Komponenten werden im Designmodus visuell wiedergegeben, wenn Sie sie der Anwendung hinzufügen.

Bibliotheken für Anwendungen konfigurieren

Für die Verwendung von SWC-Bibliotheken in Projekten haben Sie folgende Möglichkeiten:

Mit der Anwendung zusammengeführt Wenn Sie eine SWC-Datei zum Bibliothekspfad des Projekts hinzufügen, können Sie die in der Bibliothek enthaltenen Komponenten in der Anwendung verwenden. Wenn Sie die Anwendung erstellen, werden nur die Bibliothekskomponenten, die Sie tatsächlich verwendet haben, in der SWF-Anwendungsdatei kompiliert. Anders ausgedrückt: Der komplette Anwendungscode wird in einer SWF-Datei zusammengeführt. Dies ist die häufigste und einfachste Methode, Bibliothekskomponenten zu verwenden.

Außerhalb der Anwendung Sie können Bibliothekskomponenten separat von der kompilierten SWF-Datei halten, damit sie nicht in der Datei zusammengeführt werden. Der Compiler löst den gesamten Code der Bibliothek auf, die von der Anwendung verwendet wird, führt den Code aber nicht in der SWF-Anwendungsdatei zusammen. Der Vorteil dieses Ansatzes ist, dass die SWF-Anwendungsdatei hierbei kleiner wird. Die in der SWC-Datei enthaltenen Komponenten werden zur Laufzeit bei Bedarf abgerufen und in den Speicher geladen.

Runtime Shared Library Bei Flex-Projekten können Sie SWC-Dateien auch als Runtime Shared Library (RSL) nutzen, die einer Dynamic Link Library auf anderen Plattformen ähnlich ist. Verwenden Sie SWC-Dateien als RSL, wenn Sie eine Sammlung von Komponenten haben, die von mehr als einer Anwendung verwendet werden.

Die gemeinsame Nutzung von Komponenten in Anwendungen über eine RSL bietet mehrere Vorteile. Erstens wird die Bibliothek nur einmal in den Speicher geladen, zwischengespeichert und steht dann allen Anwendungen zur Verfügung, die diese Komponenten nutzen. Zweitens werden die in der Bibliothek enthaltenen Komponenten nur geladen, wenn sie benötigt werden. Die einzelnen Anwendungen sind hierdurch kleiner und können somit schneller aufgerufen werden. Das potenzielle Problem bei diesem Ansatz ist, dass die gesamte RSL in den Speicher geladen wird, anstatt der einzelnen Komponenten, die von den Anwendungen verwendet werden. Weitere Informationen zum Verwenden von SWC-Dateien als RSL finden Sie unter Runtime Shared Libraries.

Flex-Bibliotheksprojekte erstellen

Wenn Sie ein Bibliotheksprojekt erstellen, führt der Assistent „Neues Flex-Bibliotheksprojekt“ Sie durch die einzelnen Schritte und fordert Sie auf, den Projektnamen, den Pfad und Informationen zum Erstellungspfad anzugeben.

Der erste Schritt beim Erstellen einer SWC-Datei in Flash Builder besteht darin, ein Flex-Bibliotheksprojekt zu erstellen. Nachdem Sie das Bibliotheksprojekt erstellt haben, fügen Sie Komponenten hinzu, geben die Bibliotheksprojektelemente an, die in der SWC-Datei enthalten sein sollen, und erstellen dann das Projekt, um die SWC-Datei zu generieren.

1 Wählen Sie „Datei“ > „Neu“ > „Flex-Bibliotheksprojekt“ aus.

2 Geben Sie einen Projektnamen ein und geben Sie dann Folgendes an:

Projektpfad Der Standardspeicherort ist der aktuelle Arbeitsbereich. Auf Windows-Plattformen ist der Speicherort des Standardarbeitsbereichs C:\Dokumente und Einstellungen*Benutzername*\Adobe Flash Builder\. Unter Mac OS ist der Speicherort des Standardarbeitsbereichs /Benutzer/*Benutzername*/Adobe Flash Builder/. Sie können einen anderen Projektpfad wählen, indem Sie die Option „Standardpfad verwenden“ deaktivieren.

Konfiguration Sie können angeben, ob das Flex-Bibliotheksprojekt generische Bibliotheken oder Mobilbibliotheken verwendet. Generische Bibliotheken werden für Web-, Desktop- und Mobilprojekte verwendet.

Weiterhin können Sie angeben, ob das Flex-Bibliotheksprojekt mit Flash Catalyst kompatibel sein muss. In diesem Fall werden generische Bibliotheken verwendet. Mobilbibliotheken werden für mit Flash Catalyst kompatible Projekte nicht unterstützt.

Flex SDK-Version Aktivieren Sie die Option „Standard SDK verwenden“ oder „Bestimmtes SDK verwenden“. Sie können auch auf den Link „Flex SDKs konfigurieren“ klicken, um SDKs auf der Hauptvoreinstellungsseite hinzuzufügen, zu bearbeiten oder zu entfernen.

Adobe AIR-Bibliotheken einschließen Aktivieren Sie diese Option, wenn die Bibliothek AIR-Funktionen wie z. B. Zugriff auf die AIR-APIs verwenden muss. Flash Builder ändert dann den Bibliothekspfad dieses neuen Flex-Bibliotheksprojekts so, dass es „airglobal.swc“ und „airframework.swc“ aufnimmt. Webbasierte Flex-Projekte können diese Bibliothek nicht verwenden.

Aktivieren Sie diese Option nicht, wenn Sie eine generische Bibliothek schreiben, die nur in einer webbasierten Anwendung oder in einer web- oder AIR-basierten Anwendung verwendet werden soll.

3 Klicken Sie auf „Weiter“.

4 (Optional) Legen Sie Informationen zum Erstellungspfad fest. Sie können beispielsweise Ordner zum Projektquellpfad, der die in die SWC-Datei aufzunehmenden Komponenten enthält, hinzufügen. Sie können auch andere Projekte, Ordner oder SWC-Bibliotheksdateien hinzufügen, die in das Bibliotheksprojekt aufgenommen werden sollen. Siehe „SWC-Dateien in Projekten verwenden“ auf Seite 237.

5 Wenn Sie alle Einstellungen für das Projekt vorgenommen haben, klicken Sie auf „Fertig stellen“.

Eine bewährte Vorgehensweise für die Erstellung von Flex-Bibliotheken finden Sie im Artikel Three points to remember when creating a library von Adobe Flex-Experte Xavi Beumala.

AIR-Bibliotheksprojekt erstellen

Um eine AIR-Codebibliothek für mehrere AIR-Projekte zu erstellen, erstellen Sie mit dem Standardassistenten für Flex-Bibliotheksprojekte ein AIR-Bibliotheksprojekt.

1 Wählen Sie „Datei“ > „Neu“ > „Flex-Bibliotheksprojekt“ aus.

2 Geben Sie einen Projektnamen ein.

3 Aktivieren Sie die Option „Adobe AIR-Bibliotheken einschließen“ und klicken Sie dann auf „Weiter“

4 Ändern Sie den Erstellungspfad bei Bedarf ab und klicken Sie dann auf „Fertig stellen“. Weitere Informationen zum Erstellen von Bibliotheksprojekten finden Sie unter „Bibliotheksprojekte“ in der Flash Builder-Hilfe.

Komponenten zum Bibliotheksprojekt hinzufügen

Für das Hinzufügen von Komponenten zum Bibliotheksprojekt haben Sie folgende Möglichkeiten:

- Fügen Sie zum Projekt neue oder vorhandene benutzerdefinierte Komponenten, ActionScript-Klassen und andere Elemente hinzu.
- Verlinken Sie vorhandene Komponenten in anderen Projekten mit dem Arbeitsbereich. (Siehe „Mit Ressourcen außerhalb des Projektarbeitsbereichs verlinken“ auf Seite 46.)
- Fügen Sie zum Quellpfad des Bibliotheksprojekts einen verlinkten Ordner, der Komponenten enthält, hinzu. (Siehe „Externen Ressourcenordner zum Quellpfad hinzufügen“ auf Seite 47.)

Hinweis: *Alle Komponenten, die Sie in das Bibliotheksprojekt aufnehmen, müssen mit dem Bibliotheksprojekt (direkt oder als verlinkte Ressourcen) verknüpft sein.*

Bibliotheksprojektelemente zur Aufnahme in die SWC-Datei auswählen

Gehen Sie wie folgt vor, um die Elemente (Komponenten und Ressourcen) für die SWC-Datei auszuwählen:

1 Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex Bibliothek-Erstellungspfad“.

Die Komponenten, die Sie zum Projekt (entweder direkt oder durch Verlinkung) hinzugefügt haben, werden auf der Registerkarte „Klassen“ angezeigt.

2 Wählen Sie die Komponentenklassen aus, die in die SWC-Datei aufgenommen werden sollen.

3 (Optional) Klicken Sie auf die Registerkarte „Ressourcen“ und wählen Sie die Ressourcen aus, die in die SWC-Datei aufgenommen werden sollen.

4 Wenn Sie alle gewünschten Elemente ausgewählt haben, klicken Sie auf „OK“.

Bibliotheksprojekte erstellen

Nachdem Sie Elemente ausgewählt haben, die in die SWC-Datei aufgenommen werden sollen, und sofern Sie die Option „Automatisch erstellen“ aktiviert haben, wird die SWC-Datei sofort kompiliert und im Ausgabeordner des Projekts generiert. Wenn Sie Projekte manuell erstellen, können Sie das Bibliotheksprojekt zum gewünschten Zeitpunkt über „Projekt“ > „Projekt erstellen“ bzw. „Alle erstellen“ erstellen.

Beim Erstellen des Bibliotheksprojekts wird eine SWC-Datei generiert, die Sie in anderen Anwendungen nutzen bzw. an andere Benutzer weitergeben können.

Eine SWC-Datei ist eine Archivdatei. Sie können die SWC-Datei in jedem Archivprogramm wie z. B. WinZip öffnen. In der SWC-Datei befinden sich die Dateien „library.swf“ und „catalog.xml“. Außerdem sind Eigenschaftendateien und andere eingebettete Elemente enthalten.

Sie können die Bibliothek als Open Directory anstatt als SWC-Datei exportieren. Sie exportieren eine Bibliothek typischerweise als Open Directory, wenn Sie vorhaben, die Datei „library.swf“ in der SWC-Datei als RSL zu verwenden.

Hierzu legen Sie die Compiler-Optionen für `directory` und `output` fest. Sie setzen die Option `output` auf den Namen des anzulegenden Ordners und die Option `directory` auf `true`, um anzugeben, dass Sie ein Open Directory und keine SWC-Datei haben möchten, wenn Sie die Bibliothek erstellen. Um die Compiler-Optionen zu bearbeiten, wählen Sie „Projekt“ > „Eigenschaften“ > „Flex Bibliothek-Compiler“ aus und fügen Sie die Optionen zum Feld „Zusätzliche Compiler-Argumente“ hinzu, z. B.:

```
-directory=true -output=myOpenDir
```

Flash Builder legt im Projekt einen Ordner mit dem Namen „myOpenDir“ an und speichert den Inhalt der SWC-Datei in diesem Ordner.

SWC-Dateien in Projekten verwenden

Um SWC-Dateien in Flex-Projekten zu verwenden, fügen Sie sie zum Bibliothekspfad des Projekts hinzu. Die SWC-Dateien können im Projekt, in einem Flex-Bibliotheksprojekt, in einem gemeinsam genutzten Ordner im Arbeitsbereich oder in jedem anderen Pfad abgelegt sein, der mit dem Projekt verlinkt wurde (z. B. in einem gemeinsam genutzten Ordner, der zum Quellpfad des Projekts hinzugefügt wurde).

Wenn Sie in Anwendungen SWC-Dateien verwenden, wird über Konfigurationsoptionen festgelegt, ob die Dateien statisch oder dynamisch mit der Anwendung verlinkt sind, ob sie in der SWF-Anwendungsdatei zusammengeführt wurden oder außerhalb von ihr liegen und zur Laufzeit separat auf sie zugegriffen wird.

SWC-Datei zum Bibliothekspfad hinzufügen

1. Wählen Sie im Paket-Explorer ein Projekt aus und klicken Sie auf „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.
2. Klicken Sie auf die Registerkarte „Bibliothekspfad“.
3. Wählen Sie eine der folgenden Optionen aus, um SWC-Dateien hinzuzufügen:

   Projekt hinzufügen Fügt ein Flex-Bibliotheksprojekt hinzu.

   SWC-Ordner hinzufügen Fügt einen Ordner hinzu, der SWC-Dateien enthält.

   SWC hinzufügen Fügt eine kompilierte SWC-Datei hinzu.

   Flex SDK hinzufügen Fügt andere Flex SDKs hinzu. Wenn das Projekt bereits ein Flex SDK im Bibliothekspfad enthält, ist diese Schaltfläche deaktiviert. Wenn Sie das vorhandene Flex SDK aus dem Bibliothekspfad entfernen, wird die Schaltfläche aktiviert. Wenn Sie auf diese Schaltfläche klicken, wird ein Flex SDK-Knoten hinzugefügt, das Programm teilt Ihnen aber nicht mit, welches Flex SDK hinzugefügt wurde. Um zu steuern, welches Flex SDK verwendet wird, wählen Sie „Projekt“ > „Eigenschaften“ > „Flex Compiler“.
4. Geben Sie den Pfad der SWC-Datei, des Projekts oder des Ordners ein oder wählen Sie die SWC-Datei, das Projekt oder den Ordner aus. Klicken Sie auf „OK“.

   Die SWC-Datei, das Bibliotheksprojekt oder der Ordner wird zum Bibliothekspfad hinzugefügt.

SWC-Datei in der SWF-Anwendungsdatei zusammenführen, wenn sie kompiliert ist

1. Wählen Sie im Paket-Explorer ein Projekt aus und klicken Sie auf „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.
2. Klicken Sie auf die Registerkarte „Bibliothekspfad“ und wählen Sie dann den Eintrag „SWC-Datei“ und erweitern Sie ihn, um die SWC-Optionen anzuzeigen.
3. Doppelklicken Sie auf die Option „Verbindungstyp“. Das Dialogfeld mit den Optionen für Bibliothekspfadelemente wird angezeigt.
4. Aktivieren Sie die Option „In Code zusammengeführt“ und klicken Sie auf „OK“.

Diese Vorgehensweise entspricht der Compiler-Option `library-path`.

SWC-Datei als externe Bibliotheksdatei festlegen

1. Wählen Sie im Paket-Explorer ein Projekt aus und klicken Sie auf „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.
2. Klicken Sie auf die Registerkarte „Bibliothekspfad“ und wählen Sie dann den Eintrag „SWC-Datei“ aus und erweitern Sie ihn, um die SWC-Optionen anzuzeigen.
3. Doppelklicken Sie auf die Option „Verbindungstyp“. Das Dialogfeld mit den Optionen für Bibliothekspfadelemente wird angezeigt.
4. Aktivieren Sie die Option „Extern“ und klicken Sie auf „OK“.

Diese Vorgehensweise entspricht der Compiler-Option `external-library-path`.

SWC-Datei als RSL verwenden

Sie können die gemeinsam genutzten Elemente in der SWF-Datei der Anwendung in eigenständige Dateien auslagern. Diese eigenständigen Dateien können separat auf den Client heruntergeladen und zwischengespeichert werden. Die gemeinsam genutzten Elemente werden nur einmal auf den Client übertragen und geladen, aber eine beliebige Anzahl von Anwendungen kann sie zur Laufzeit verwenden. Diese gemeinsam genutzten Dateien werden als *Runtime Shared Libraries* oder *RSLs* bezeichnet.

1. Wählen Sie im Paket-Explorer ein Projekt aus und klicken Sie auf „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“.
2. Klicken Sie auf die Registerkarte „Bibliothekspfad“ und wählen Sie dann den Eintrag „SWC-Datei“ aus und erweitern Sie ihn, um die SWC-Optionen anzuzeigen.
3. Doppelklicken Sie auf die Option „Verbindungstyp“ oder klicken Sie auf „Bearbeiten“. Das Dialogfeld mit den Optionen für Bibliothekspfadelemente wird angezeigt.
4. Wählen Sie „Runtime Shared Library (RSL)“ als Verbindungstyp aus.
5. Sie können eine URL hinzufügen, um beim Bereitstellen der Anwendung den Speicherort der SWC-Bibliothek anzugeben. Bei Bedarf können Sie auch eine vorhandene URL bearbeiten.
6. Wählen Sie „Laden von RSL erzwingen“ aus, um während der Kompilierung das Laden einer nicht benutzten RSL zu erzwingen.
7. Sie können die Anwendungsdomäne angeben, in der die domänenübergreifenden RSLs geladen werden müssen.

Die Verwendung von SWC-Dateien als RSL vereinfacht den Prozess zur manuellen Verwendung von RSLs. Extrahieren Sie hierzu die SWF-Datei aus der SWC-Datei und legen Sie die Werte der Compiler-Option `runtime-shared-library-path` fest.

Weitere Informationen zum Verwenden von SWC-Dateien als RSL finden Sie unter Runtime Shared Libraries in *Verwenden von Adobe Flex*.

Benutzerdefinierte MXML-Komponenten erstellen

Sie können benutzerdefinierte Komponenten erstellen, um eine vorhandene Komponente zu erweitern. Oder Sie erstellen eine wiederverwendbare Komponente, wie z. B. ein Suchfeld oder die Anzeige eines Elements in einem Datenraster. Sie können auch eine neue Komponente programmieren, die im Flex-Framework nicht verfügbar ist.

Die von Ihnen erstellten Komponenten werden im Designmodus in der Komponentenansicht angezeigt. Sie können die Komponenten mithilfe von SWC-Dateien verteilen. Weitere Informationen finden Sie unter „SWC-Dateien in Projekten verwenden“ auf Seite 237.

Wenn die Komponente zum Großteil aus vorhandenen Komponenten besteht, bietet es sich an, sie mit MXML zu definieren. Handelt es sich aber um einen völlig neuen Komponententyp, sollten Sie sie als ActionScript-Komponente definieren. Weitere Informationen hierzu finden Sie unter „ActionScript-Klasse erstellen“ auf Seite 43.

1 Wählen Sie „Datei“ > „Neu“ > „MXML-Komponente“.

Das Dialogfeld „MXML-Komponente - Neu“ wird angezeigt.

2 Legen Sie den übergeordneten Ordner für die benutzerdefinierte Komponentendatei fest.

Speichern Sie die Datei in einem Ordner im aktuellen Projektordner oder im Quellpfad des aktuellen Projekts, wenn die Komponente in der Komponentenansicht angezeigt werden soll.

3 Geben Sie einen Dateinamen für die Komponente an.

Der Dateiname definiert den Komponentennamen. Wenn Sie als Namen für die Datei z. B. `LoginBox.mxml` eingeben, wird als Name für die Komponente „LoginBox“ verwendet.

4 Wählen Sie im Feld „Basiert auf“ die Grundkomponente für die benutzerdefinierte Komponente aus.

Benutzerdefinierte Komponenten werden typischerweise von vorhandenen Komponenten abgeleitet. Als Grundkomponenten für benutzerdefinierte Layoutkomponenten werden häufig Container verwendet.

Ab Flex 4 schlägt Flash Builder **spark.components.Group** als Grundkomponente vor.

Wählen Sie „Durchsuchen“, um das Dialogfenster „Typ öffnen“ zu öffnen und eine Komponente zu wählen.

Modifizieren oder löschen Sie die vorgeschlagene Komponente, um die Auswahl im Dialogfenster „Typ öffnen“ zu erweitern. Geben Sie z. B. **spark.components.** an und klicken Sie dann auf „Durchsuchen“.

Sie können im Dialogfenster „Typ öffnen“ die Auswahl Ihren Anforderungen entsprechend filtern. Informationen zur Verwendung des Dialogfensters „Typ öffnen“ finden Sie unter „Klassen durchsuchen und anzeigen“ auf Seite 85.

5 (Optional) Wenn die Komponente auf einem Container basiert, werden Optionen zum Festlegen der Breite und Höhe der Komponente angezeigt.

Sie können diese Optionen auf eine feste Breite und Höhe oder auf Prozentwerte setzen, oder aber die Werte löschen. Wenn Sie eine Instanz der Komponente erstellen, können Sie die Breite und Höher der Komponente in der Instanz überschreiben.

Wenn Sie für die Breite und Höhe einen Prozentsatz eingeben oder keine Breite und Höhe festlegen, können Sie sich in einer Vorschau ansehen, wie die Komponente in verschiedenen Größen aussieht. Arbeiten Sie hierzu im Designmodus des MXML-Editors und wählen Sie die verschiedenen Größen über das Popupmenü „Designbereich“ in der Symbolleiste des MXML-Editors. Weitere Informationen hierzu finden Sie unter „Komponenten visuell entwerfen“ auf Seite 240.

6 Klicken Sie auf „Fertig stellen“.

Flash Builder speichert die Datei im übergeordneten Ordner und öffnet sie im Editor.

Wenn Sie die Datei im aktuellen Projekt oder im Quellpfad des aktuellen Projekts gespeichert haben, zeigt Flash Builder auch die Komponente in der Komponentenansicht an, sodass Sie diese schnell in Anwendungen einfügen können. Weitere Informationen hierzu finden Sie unter „Komponenten im MXML-Designmodus hinzufügen“ auf Seite 24.

Hinweis: *In der Komponentenansicht werden nur sichtbare benutzerdefinierte Komponenten (Komponenten, die von der Klasse „UIComponent“ erben) angezeigt. Weitere Informationen finden Sie im ActionScript 3.0 Referenzhandbuch für die Adobe Flash-Plattform.*

7 Erstellen Sie die benutzerdefinierte Komponente.

Weitere Informationen hierzu finden Sie unter Einfache MXML-Komponenten.

Komponenten visuell entwerfen

Das Layout für benutzerdefinierte Komponenten wird im MXML-Editor genau so wie das für eine normale MXML-Anwendungsdatei entwickelt. Alle Werkzeuge im Designmodus stehen zur Verfügung. Beispielsweise können Sie aus der Komponentenansicht untergeordnete Steuerelemente hinzufügen und dann in der Eigenschaftenansicht Einstellungen festlegen. Weitere Informationen finden Sie unter „Visuell mit Komponenten im MXML-Designmodus arbeiten“ auf Seite 23.

Folgende Regeln gelten für die Größe der benutzerdefinierten Komponente:

- Wenn die Komponente eine feste Breite und Höhe hat, stellt Flash Builder den Designbereich automatisch auf diese Breite und Höhe ein.
- Wenn die Komponente keine Breite und Höhe oder eine Breite und Höhe von 100% hat, können Sie die Größe des Designbereichs im Popupmenü „Designbereich“ in der Symbolleiste des Editors auswählen.

  Wenn Sie eine andere Größe auswählen, können Sie die Komponente so in der Vorschau anzeigen, wie sie in verschiedenen Situationen erscheinen würde. Die Größe des Designbereichs wird für Container standardmäßig auf 400 x 300 Pixel und für Steuerelemente standardmäßig auf „An Inhalt anpassen“ gesetzt.
- Wenn die Komponente auf einen anderen Prozentwert als 100% für die Breite und Höhe eingestellt ist, gibt Flash Builder die Komponente als Prozentsatz der ausgewählten Größe im Menü „Designbereich“ wieder.

ASDoc-Kommentare für benutzerdefinierte Komponenten bereitstellen

Sie können die benutzerdefinierten Komponenten dokumentieren, indem Sie zum Code, der die Komponenten implementiert, ASDoc-Kommentare hinzufügen. ASDoc-Kommentare stehen anschließend mit der Inhaltshilfe im MXML- und ActionScript-Editor zur Verfügung. Weitere Informationen hierzu finden Sie unter „Inhaltshilfe“ auf Seite 49.

Fügen Sie ActionScript-Quelldateien ASDoc-Kommentare hinzu, um API-Referenzdokumentation zur Verfügung zu stellen. Sie können auch ASDoc-Kommentare hinzufügen, um MXML-Elemente zu dokumentieren. Weitere Informationen zum Erstellen von ASDoc-Kommentaren für die Quelldateien finden Sie unter ASDoc.

Benutzerdefinierte MXML-Komponenten bearbeiten und verteilen

Flash Builder zeigt alle sichtbaren benutzerdefinierten Komponenten (Komponenten, die von „UIComponent“ erben) im Designmodus des MXML-Editors an. Sie können auf eine benutzerdefinierte Komponente im Layout doppelklicken, um die dazugehörige Datei zu öffnen und zu bearbeiten.

1 Öffnen Sie eine MXML-Datei, die eine benutzerdefinierte Komponente verwendet.

2 Doppelklicken Sie im Designmodus auf eine benutzerdefinierte Komponente im Layout.

Die Komponentendatei wird im Editor geöffnet.

Sie können auch im Kontextmenü „Benutzerdefinierte Komponente öffnen“ wählen.

3 Bearbeiten Sie die benutzerdefinierte Komponente.

Sie verteilen benutzerdefinierte Komponenten, indem Sie Bibliotheksprojekte erstellen. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 234.

Module erstellen

In Flash Builder können Sie Module erstellen, hinzufügen, optimieren und debuggen. Informationen zum Schreiben von Modulcode finden Sie unter Modulare Anwendungen.

Die folgenden Schritte beschreiben, wie in Flash Builder ein Modul erstellt wird. Nachdem Sie ein Modul erstellt haben, können Sie es kompilieren.

1 Wählen Sie in Flash Builder die Option „Datei“ > „Neu“ > „MXML-Modul“. Das Dialogfeld zum Erstellen eines neuen MXML-Moduls wird angezeigt.

2 Wählen Sie einen übergeordneten Ordner für das Modul aus. Module werden typischerweise im selben Ordner wie die Hauptanwendung gespeichert, damit die relativen Pfade zu gemeinsam genutzten Ressourcen identisch sind.

Da Module ausführbar sind, müssen sie sich im Quellordner des Projekts befinden.

3 Geben Sie einen Dateinamen für das Modul ein, beispielsweise „MeinModul“.

4 Geben Sie die Breite, Höhe und Layouteigenschaften für das Modul ein.

5 Geben Sie an, ob das Modul optimiert werden soll.

Wenn Sie ein Modul einer Anwendung optimieren, werden Klassen, die von der Anwendung verwendet werden, vom Modul ausgeschlossen. Hierdurch können kleinere Downloadgrößen der SWF-Dateien erreicht werden. Weitere Informationen finden Sie unter „Module in Flash Builder optimieren“ auf Seite 245.

- Für Anwendung optimieren

  Wenn Sie diese Option wählen, geben Sie eine Anwendung an, für die das Modul optimiert werden soll.

- Nicht optimieren

  Wenn Sie diese Option wählen, werden alle Klassen in das Modul eingeschlossen, unabhängig davon, ob sie in der Hauptanwendung definiert sind. Hierdurch kann die Leistung der inkrementellen Kompilierung erhöht werden. Zusätzlich können Sie das Modul in jede beliebige Anwendung laden, da alle seine Abhängigkeiten im Modul kompiliert werden.

6 Klicken Sie auf „Fertig stellen“.

Flash Builder fügt zum Projekt eine neue MXML-Moduldatei hinzu.

Module kompilieren

In Flash Builder können Sie ein Modul entweder so ausführen, als wäre es eine Anwendung, oder das Projekt des Moduls erstellen. Wenn sich die Module im selben Projekt wie Ihre Anwendung befinden, kompiliert Flash Builder beim Ausführen der Anwendung SWF-Dateien für alle Module des Projekts. Die SWF-Dateien werden dann zur Laufzeit in die Anwendung geladen.

Eine modulbasierte SWF-Datei kann nicht als eigenständige SWF-Datei ausgeführt oder in ein Browserfenster geladen werden. Sie muss von einer Anwendung als Modul geladen werden. Wenn Sie das Modul in Flash Builder ausführen, um es zu kompilieren, sollten Sie Adobe Flash Player bzw. den Browser schließen und eventuelle Fehlermeldungen ignorieren. Module sollten nicht direkt durch den Player oder Browser angefordert werden.

Die SWF-Moduldateien und die SWF-Datei der Hauptanwendung befinden sich normalerweise im selben Ordner. Flash Builder kompiliert jedoch die Module zur selben Zeit wie Ihre Anwendung, unabhängig von ihrem Speicherort. Module können sich im selben Ordner befinden wie die Anwendung oder in Unterordnern.

Sie können auch ein separates Flex- oder ActionScript-Projekt für jedes Modul oder jede Modulgruppe erstellen. Hierdurch erhalten Sie mehr Kontrolle darüber, wie Module kompiliert werden, da jedes Projekt über andere Compiler-Optionen als die Anwendung oder andere Module verfügen kann. Sie können auch das bzw. die Projekte des Moduls kompilieren, ohne die Anwendung zu kompilieren. Bei dieser Vorgehensweise müssen Sie jedoch jedes Modul manuell kompilieren, bevor Sie die Anwendung kompilieren. Eine Möglichkeit besteht darin, alle geöffneten Projekte in Flash Builder in einem Arbeitsgang zu kompilieren.

Wenn Sie Module separat von der Hauptanwendung kompilieren, müssen Sie (je nachdem, ob Sie die Anwendung und Module debuggen möchten oder nicht) festlegen, ob Debuginformationen mit aufgenommen oder ausgeschlossen werden sollen. Weitere Informationen finden Sie unter „Module in Flash Builder debuggen“ auf Seite 246.

Das Funktionsprinzip in Flash Builder basiert auf der Verknüpfung von Modulen mit einer einzigen Anwendung. Sollen Module in mehreren Anwendungen verwendet werden, können Sie den Code in eine Bibliothekskomponente oder -klasse einbinden und diese in ein einfaches Modul für jede Anwendung aufnehmen. Module sind nicht für die Wiederverwendung von Code in verschiedenen Anwendungen gedacht. Diese Rolle übernehmen Bibliotheken.

Mehrere Projekte für Module verwenden

Beim Einrichten der Projektarchitektur können Sie entscheiden, ob Sie Module in das Anwendungsprojekt aufnehmen, separate Projekte für die einzelnen Module oder ein separates Projekte für alle Module erstellen möchten.

Ein Projekt pro Modul verwenden

Vorteile, wenn je Modul ein Projekt verwendet wird:

- Modulprojekte können an einem beliebigen Ort im Arbeitsbereich gespeichert sein.
- Modulprojekte können ihre eigenen Compiler-Einstellungen, z. B. einen benutzerdefinierten Bibliothekspfad, haben.

  Modulprojekte können die Compiler-Option `load-externs` verwenden, um überlappende Abhängigkeiten zu entfernen.

Nachteile, wenn je Modul ein Projekt verwendet wird:

- Viele Projekte verbrauchen mehr Speicherplatz.
- Viele Projekte im Arbeitsbereich können dazu führen, dass dieser unübersichtlich wird.
- Standardmäßig werden beim Kompilieren der Anwendung nicht alle Modulprojekte kompiliert, selbst dann nicht, wenn sie sich geändert haben.

- Um die Dateigröße des Moduls zu optimieren, wenden Sie die Compiler-Optionen `load-externs` und `link-report` manuell an.

Ein Projekt für alle Module verwenden

Eine weitere Möglichkeit besteht darin, ein einziges Projekt für alle Module zu verwenden, die Anwendung aber in ihrem eigenen separaten Projekt zu belassen. Dies hat einige der Nachteile, die bei der Verwendung eines einzigen Projekts für die Anwendung und die Module auftreten, aber viele der Vorteile, die das Verwenden je eines separaten Projekts pro Modul hat.

Vorteile, wenn ein Projekt für alle Module verwendet wird:

- Das Modulprojekt kann an einem beliebigen Ort im Arbeitsbereich gespeichert sein.
- Sie können nur die Module bzw. nur die Anwendung kompilieren, müssen also nicht beide gleichzeitig erneut kompilieren.
- Das Modulprojekt kann die Compiler-Option `load-externs` verwenden, um überlappende Abhängigkeiten zu entfernen.

Nachteile, wenn ein Modulprojekt für alle Module verwendet wird:

- Alle Module im Modulprojekt müssen dieselben Compiler-Einstellungen, z. B. den Bibliothekspfad, verwenden.
- Standardmäßig wird beim Kompilieren der Anwendung das Modulprojekt nicht kompiliert, auch dann nicht, wenn es sich geändert hat.
- Um die Dateigröße des Moduls zu optimieren, wenden Sie die Compiler-Optionen `load-externs` und `link-report` manuell an.

Projekte für Module erstellen

Beim Erstellen eines separaten Projekts für Module ändern Sie den Ausgabeordner des Modulprojekts in einen von der Anwendung verwendeten Ordner. Sie unterdrücken auch das Generieren von Wrapper-Dateien.

Separates Projekt für Module in Flash Builder erstellen

1. Erstellen Sie ein Hauptprojekt.
2. Erstellen Sie ein Projekt für ein bzw. mehrere Module.
3. Wählen Sie im Kontextmenü des Projekt des Moduls „Eigenschaften“. Das Dialogfeld „Eigenschaften“ wird angezeigt.
4. Wählen Sie die Option „Flex-Erstellungspfad“.
5. Wählen Sie als Ausgabeordner den Ordner aus, in dem sich die Module des Hauptprojekts befinden. Ändern Sie den Ausgabeordner beispielsweise wie folgt:

   ```
   ${DOCUMENTS}\MainProject\assets
   ```

   Hierdurch wird das kompilierte Modul im Elementeordner des Anwendungsprojekts (Hauptprojekt) ausgegeben. In der Hauptanwendung können Sie in der ModuleLoader `url`-Eigenschaft den Pfad zu den SWF-Dateien im Elementeordner angeben. Der Wert dieser Eigenschaft ist relativ zum Ausgabeordner.
6. Klicken Sie zum Speichern der Änderungen auf „OK“.
7. Öffnen Sie die Projekteigenschaften erneut und wählen Sie die Option „Flex Compiler“.
8. Deaktivieren Sie die Option „HTML-Wrapper-Datei generieren“. Hierdurch wird verhindert, dass vom Modulprojekt HTML-Wrapper-Dateien generiert werden. Sie verwenden diese Dateien typischerweise nur für die Anwendung. Für Module sind sie nicht notwendig.

9 Klicken Sie zum Speichern der Änderungen auf „OK“.

Projekte für Module kompilieren

In Flash Builder werden mehrere Projekte in einem Arbeitsschritt kompiliert. Wählen Sie zuerst die Reihenfolge, in der Sie die Projekte kompilieren möchten. Dann komplilieren Sie alle Projekte gleichzeitig.

Alle Projekte in Flash Builder gleichzeitig kompilieren

❖ Wählen Sie im Hauptmenü „Projekt“ > „Alle erstellen“.

Flex erstellt alle Projekte im Arbeitsbereich. Die Anwendungsdateien werden zum Ausgabeordner jedes Projekts hinzugefügt. Falls Sie nicht angegeben haben, dass die Dateien vor dem Erstellungsprozess automatisch gespeichert werden sollen, werden Sie zum Speichern der Dateien aufgefordert.

Wenn Sie die Erstellungsreihenfolge ändern möchten, verwenden Sie hierzu das Dialogfeld „Reihenfolge für Erstellung“. Es ist nicht immer erforderlich, die Erstellungsreihenfolge zu ändern. Projekte, die Module verwenden, müssen erst kompiliert werden, wenn die Hauptprojektanwendung ausgeführt wird, also nicht, wenn die Hauptprojektanwendung kompiliert wird. In den meisten Fällen kann die Standarderstellungsreihenfolge verwendet werden.

Wenn Sie allerdings überlappende Abhängigkeiten entfernen möchten, müssen Sie u. U. die Erstellungsreihenfolge ändern, damit die Hauptanwendung zuerst kompiliert wird. In diesem Fall verwenden Sie die Compiler-Option `link-report`, um den Linker-Bericht zu generieren. Beim Kompilieren von Modulen verwenden Sie die Compiler-Option `load-externs`, um den von der Shell-Anwendung generierten Linker-Bericht zu verwenden. Weitere Informationen zum Verringern der Modulgröße finden Sie unter „Module in Flash Builder optimieren“ auf Seite 245.

Erstellungsreihenfolge von Projekten ändern

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Arbeitsbereich“ > „Reihenfolge für Erstellung“ aus.

Das Dialogfeld „Reihenfolge für Erstellung“ wird angezeigt.

2 Deaktivieren Sie die Option „Standarderstellungsreihenfolge verwenden“.

3 Ändern Sie die Reihenfolge der Projekte in der Liste „Projekterstellungsreihenfolge“ über die Schaltfläche „Nach oben“ bzw. „Nach unten“. Über die Schaltfläche „Projekt entfernen“ können Sie Projekte entfernen, die nicht Teil der Hauptanwendung sind, bzw. die nicht von der Anwendung verwendete Module sind. Das entfernte Projekt wird erstellt, allerdings erst, nachdem alle Projekte in der Liste „Projekterstellungsreihenfolge“ kompiliert wurden.

4 Klicken Sie auf „OK“.

5 Ändern Sie die Erstellungsreihenfolge wie gewünscht ab und klicken Sie dann auf „OK“.

Wenn Sie im Arbeitsbereich Abhängigkeiten zwischen separaten Projekten erstellen, ermittelt der Compiler automatisch die Reihenfolge, in der die Projekte erstellt werden, damit diese Abhängigkeiten richtig aufgelöst werden.

Wenn für jedes Modul ein separates Projekt verwendet wird, können Module nacheinander kompiliert werden. Hierdurch kann Zeit gespart werden, weil in diesem Fall nicht alle Projekte auf einmal bzw. weil nicht alle Modul- und Anwendungsdateien in einem Projekt kompiliert werden müssen.

Ein aus einem Modul bestehendes Projekt kompilieren

1 Klicken Sie im Modulprojekt mit der rechten Maustaste auf die MXML-Datei des Moduls.

2 Wählen Sie „Anwendung ausführen“. Das Player- oder Browserfenster versucht, das Modul auszuführen, nachdem es kompiliert wurde. Sie können das Player- oder Browserfenster schließen und zur Laufzeit auftretende Fehlermeldungen ignorieren. Module können nicht direkt im Player oder Browser ausgeführt werden.

Module zum Projekt hinzufügen

In manchen Fällen verwenden Sie Module, die sich nicht im Hauptanwendungsprojekt befinden. Module können sich ggf. in separaten Projekten befinden, sodass Sie benutzerdefinierte Konfigurationsoptionen verwenden können. Oder das Modul soll ggf. in mehreren Anwendungen verwendet werden können. Um ein Modul im Projekt verwenden zu können, fügen Sie dem Quellpfad der Anwendung den Modulquellcode hinzu. Anschließend fügen Sie das Modul der Modulliste der Anwendung hinzu.

Ein bereits kompiliertes Modul zum Projekt hinzufügen

1. Wählen Sie im Flex Paket-Explorer die Hauptanwendung aus.
2. Wählen Sie „Projekt“ > „Eigenschaften“ > „Flex-Erstellungspfad“ aus, um den Modulquellcode dem Quellpfad des Anwendungsprojekts hinzuzufügen.
3. Klicken Sie auf die Schaltfläche „Ordner hinzufügen“ und navigieren Sie zum Quellpfad des Moduls. Klicken Sie auf „OK“, um das Modul auszuwählen. Dieser Schritt muss für jedes externe Modul, das dem Anwendungsprojekt hinzugefügt werden soll, wiederholt werden.
4. Klicken Sie erneut auf „OK“, um die Änderungen zu speichern.
5. Klicken Sie auf „Projekt“ > „Eigenschaften“ > „Flex Module“, um das Modul zur Anwendungsmodulliste hinzuzufügen. Im Dialogfeld „Flex Module“ sind alle Module aufgeführt, die zum aktuellen Projekt hinzugefügt wurden oder sich im aktuellen Projekt befinden. Wenn Sie ein neues Projekt erstellen, ist dieses Dialogfeld leer.
6. Klicken Sie auf die Schaltfläche „Hinzufügen“. Das Dialogfeld „Modul hinzufügen“ wird angezeigt.
7. Klicken Sie auf die Schaltfläche „Durchsuchen“ oder geben Sie den Speicherort der MXML-Datei des Moduls in das Feld „Quelle“ ein. Über dieses Dialogfeld können alle Module hinzugefügt werden, die sich im Quellpfad des Projekts befinden.
8. Aktivieren Sie eine der Optionen unter „SWF-Modulgröße“, um die Moduloptimierung zu aktivieren oder zu deaktivieren. Wenn Sie die Option „Für Anwendung optimieren“ aktivieren, kompiliert Flash Builder das Modul für die ausgewählte Anwendung und schließt alle in der Hauptanwendung definierten Klassen aus. Diese können Framework- oder benutzerdefinierte Klassen umfassen. Wenn Sie diese Option aktivieren, können Sie das Modul nicht in einer anderen Anwendung verwenden, da die Liste der ausgeschlossenen Klassen anders sein kann. Weitere Informationen finden Sie unter „Module in Flash Builder optimieren“ auf Seite 245.
9. Klicken Sie zum Speichern der Änderungen auf „OK“. Flash Builder fügt die Module zur Liste verfügbarer Module im Anwendungsprojekt hinzu.

Module in Flash Builder optimieren

In Flash Builder erstellen Sie ein Modul bzw. fügen ein Modul zu einem Projekt hinzu und wählen hierbei in der Regel auch gleichzeitig die Anwendung aus, für die das Modul optimiert werden soll. Wenn Sie anschließend die Anwendung ändern, für die das Modul optimiert werden soll, oder das Modul nicht optimieren wollen, können Sie die Eigenschaften des Moduls im Projekt ändern. Weitere Informationen finden Sie unter Reducing module size.

Bei dieser Prozedur wird vorausgesetzt, dass sich das Modul und die Anwendung im selben Flash Builder-Projekt befinden. Befinden sich die Module in einem separaten Projekt, fügen Sie die Compiler-Optionen `load-externs` und `link-report` manuell hinzu.

1. Klicken Sie im Flex Paket-Explorer mit der rechten Maustaste auf das Anwendungsprojekt und wählen Sie „Eigenschaften“. Das Dialogfeld „Projekteigenschaften“ wird angezeigt.
2. Wählen Sie auf der linken Seite des Dialogfelds die Option „Flex Module“.
3. Wählen Sie das gewünschte Modul aus der Modulliste aus und klicken Sie dann auf die Schaltfläche „Bearbeiten“. Das Dialogfeld „Modul bearbeiten“ wird angezeigt.

4 Aktivieren Sie zum Deaktivieren der Optimierung unter „SWF-Modulgröße“ die Option „Nicht optimieren“.

5 Um das Modul für eine andere Anwendung zu optimieren, wählen Sie im Popupmenü „Für Anwendung optimieren“ die neue Anwendung aus.

6 Klicken Sie auf „OK“.

Um die Dateigröße eines Moduls weiter zu optimieren, können Sie die Debuginformationen entfernen. Wenn Sie in Flash Builder ein Modul erstellen, werden die Debuginformationen standardmäßig mit in das Modul aufgenommen. Durch das Entfernen der Debuginformationen kann die Größe des Moduls weiter verringert werden. Anweisungen zum Entfernen von Debuginformationen aus Modulen finden Sie unter „Module in Flash Builder debuggen“ auf Seite 246.

Module in Flash Builder debuggen

Um Module und Anwendungen zu debuggen, müssen Sie beim Kompilieren der SWF-Dateien Debuginformationen mit aufnehmen. Hierzu müssen Sie die Anwendung in Flash Builder ausführen, da die Debuginformationen standardmäßig mit aufgenommen werden. Setzen Sie in der Befehlszeile die Compiler-Option `debug` auf `true`. Der Standardwert ist `true`, aber wenn Sie ihn in einer Konfigurationsdatei deaktiviert haben, müssen Sie sicherstellen, dass Sie die Einstellung außer Kraft setzen.

Flash Builder erstellt standardmäßig eine einzelne SWF-Datei, die Debugsymbole enthält. Dadurch funktionieren sowohl „Ausführen“ als auch „Debuggen“, wenn Sie eine aus Modulen bestehende Anwendung in Flash Builder ausführen. Die Debugsymbole in der SWF-Datei eines Moduls machen die SWF-Datei jedoch größer. Um Debugsymbole vor der Verteilung auszuschließen, müssen Sie das Debuggen der Anwendungsmodule deaktivieren. Hierzu exportieren Sie die Releaseversion der Module über „Projekt“ > „Releasebuild exportieren“.

Um Debuginformationen aus SWF-Dateien in Flash Builder auszuschließen, können Sie entweder die `debug`-Option im Textfeld „Zusätzliche Compiler-Argumente“ auf `false` setzen oder die SWF-Dateien über die Option „Releasebuild exportieren“ ausgeben. Diese Option generiert eine Releaseversion (also keine Debugversion). Module, die sich im aktuellen Projekt befinden, werden auf die gleiche Weise behandelt.

Wenn Sie ein separates Projekt für Module erstellen, können Sie das Debuggen für das gesamte Projekt aktivieren oder deaktivieren, ohne die Einstellungen in der Hauptanwendung zu ändern.

Wenn Sie ein Modul debuggen möchten, müssen Sie auch die Anwendung, die das Modul lädt, debuggen. Der Flex Debugger stellt keine Verbindung zu einer Anwendung her, die keine Debuginformationen enthält, auch dann nicht, wenn die Module, die die Anwendung lädt, diese Informationen enthalten. Anders gesagt: Sie können Debuginformationen nicht aus der Anwendung ausschließen, wenn Sie das Modul, das die Anwendung lädt, debuggen möchten.

Wenn Sie Module in einer AIR-Anwendung verwenden, muss sich die SWF-Moduldatei im selben Ordner wie die SWF-Hauptanwendungsdatei oder in einem ihrer Unterordner befinden.

Flex in HTML-Anwendungen integrieren

Mit der Funktion „Ajax Bridge erstellen“ können Sie JavaScript-Code und eine HTML-Wrapper-Datei generieren, um eine in Flex aus JavaScript erstellte Anwendung problemlos in einer HTML-Seite verwenden zu können. Diese Funktion kommt zusammen mit der Flex Ajax Bridge JavaScript-Bibliothek zum Einsatz, in der Sie eine Anwendung für das Skripten im Webbrowser bereitstellen können. Der generierte JavaScript-Code ist einfach, da er zum Bereitstellen einer Funktion dient, die die Flex Ajax Bridge bereits enthält. Weitere Informationen zu Flex Ajax Bridge finden Sie unter Flex Ajax Bridge.

Die Funktion „Ajax Bridge erstellen“ generiert JavaScript-Proxycode, der für die Anwendungs-APIs, die Sie in JavaScript aufrufen möchten, spezifisch ist. Sie können Code für jede MXML-Anwendung oder ActionScript-Klasse in einem Flash Builder-Projekt generieren.

Für MXML-Anwendungsdateien können Sie Code für ein beliebiges oder alle der folgenden Elemente im MXML-Code generieren:

- Liste vererbter Elemente, die nichtrekursiv erweitert werden können
- Öffentliche Eigenschaften, einschließlich Tags mit `id`-Eigenschaften
- Öffentliche Konstanten
- Öffentliche Funktionen, einschließlich in Zeilen definierte Klassen

Für ActionScript-Klassen können Sie Code für ein beliebiges oder alle der folgenden Elemente generieren:

- Liste vererbter Elemente
- Öffentliche Eigenschaften; für jede Eigenschaft wird eine get- und eine set-Methode angezeigt
- Öffentliche Konstanten
- Öffentliche Methoden

Die Funktion „Ajax Bridge erstellen“ generiert im von Ihnen angegebenen Ordner JS- und HTML-Dateien, die den ausgewählten MXML-Anwendungen und ActionScript-Klassen entsprechen. Sie kopiert die Flex Ajax Bridge-Bibliothek (fabridge.js) in einen Unterordner des zum Generieren des Codes verwendeten Ordners. Diese Funktion erstellt außerdem MXML-Hilfsdateien im `src`-Ordner des Projekts. Diese Dateien werden zum Abschließen der JavaScript-Codegenerierung verwendet.

Ajax Bridge-Code generieren

1. Klicken Sie im Flex Paket-Explorer mit der rechten Maustaste auf ein Projekt und wählen Sie „Ajax Bridge erstellen“.
2. Wählen Sie im Dialogfeld „Ajax Bridge erstellen“ die MXML-Anwendungen und ActionScript-Klassen aus, für die Sie JavaScript-Code erstellen möchten. Sie können das Kontrollkästchen auf der obersten Ebene markieren, um das gesamte Objekt einzuschließen, oder bestimmte Mitglieder festlegen.
3. Geben Sie den Ordner an, in dem Proxyklassen generiert werden sollen.
4. Klicken Sie auf „OK“, um den Code zu generieren. Das folgende Beispiel zeigt eine JS-Datei, die für eine Anwendung zum Anzeigen von Bildern generiert wurde:

```
*
    * You should keep your JavaScript code inside this file as light as possible,
    * and keep the body of your Ajax application in separate *.js files.
    *
    * Do make a backup of your changes before regenerating this file. (Ajax Bridge
    * display a warning message.)
    *
    * For help in using this file, refer to the built-in documentation in the Ajax Bridge
application.
    *
    */


/**
    * Class "DisplayShelfList"
    * Fully qualified class name: "DisplayShelfList"
    */
function DisplayShelfList(obj) {
    if (arguments.length > 0) {
        this.obj = arguments[0];
    } else {
        this.obj = FABridge["b_DisplayShelfList"].
            create("DisplayShelfList");
    }
}

// CLASS BODY
// Selected class properties and methods
DisplayShelfList.prototype = {

    // Fields form class "DisplayShelfList" (translated to getters/setters):

    // Methods form class "DisplayShelfList":

    getAngle : function() {
        return this.obj.getAngle();
    },

    setAngle : function(argNumber) {
        this.obj.setAngle(argNumber);
    },

    setCurrentPosition : function(argNumber) {
        this.obj.setCurrentPosition(argNumber);
    },

    setSelectedIndex : function(argNumber) {
        this.obj.setSelectedIndex(argNumber);
    },

    setPercentHeight : function(argNumber) {
        this.obj.setPercentHeight(argNumber);
    },

    setPercentWidth : function(argNumber) {
        this.obj.setPercentWidth(argNumber);
    },
```

```
    DisplayShelfList : function() {
        return this.obj.DisplayShelfList();
    },

    setFirst : function(argNumber) {
        this.obj.setFirst(argNumber);
    },

    setFormat : function(argString) {
        this.obj.setFormat(argString);
    },

    setLast : function(argNumber) {
        this.obj.setLast(argNumber);
    }
}


/**
    * Listen for the instantiation of the Flex application over the bridge.
    */
FABridge.addInitializationCallback("b_DisplayShelfList", DisplayShelfListReady);


/**
    * Hook here all of the code that must run as soon as the DisplayShelfList class
    * finishes its instantiation over the bridge.
    *
    * For basic tasks, such as running a Flex method on the click of a JavaScript
    * button, chances are that both Ajax and Flex have loaded before the
    * user actually clicks the button.
    *
    * However, using DisplayShelfListReady() is the safest way, because it lets
    * Ajax know that involved Flex classes are available for use.
    */
function DisplayShelfListReady() {

    // Initialize the root object. This represents the actual
    // DisplayShelfListHelper.mxml Flex application.
    b_DisplayShelfList_root = FABridge["b_DisplayShelfList"].root();


    // YOUR CODE HERE
    // var DisplayShelfListObj = new DisplayShelfList();
    // Example:
    // var myVar = DisplayShelfListObj.getAngle ();
    // b_DisplayShelfList_root.addChild(DisplayShelfListObj);

}
```

5 Bearbeiten Sie die generierte JS-Datei. Fügen Sie in der Funktion `xxxReady()` der generierten JS-Dateien Code ein, der ausgeführt werden muss, sobald die entsprechende Klasse die Instanziierung über die Ajax Bridge beendet. Abhängig von der Anwendung kann in dieser Methode Standardcode generiert werden. Der fett formatierte Code im folgenden Beispiel zeigt benutzerdefinierten Initialisierungscode für die Bild-Beispielanwendung:

```
...
function DisplayShelfListReady() {

    // Initialize the root object. This represents the actual
    // DisplayShelfListHelper.mxml Flex application.
    b_DisplayShelfList_root = FABridge["b_DisplayShelfList"].root();


       // Create a new object.
       DisplayShelfListObj = new DisplayShelfList();
       // Make it as big as the application.
       DisplayShelfListObj.setPercentWidth(100);
       DisplayShelfListObj.setPercentHeight(100);
       //Set specific attributes.
        DisplayShelfListObj.setFirst(1);
        DisplayShelfListObj.setLast(49);
        DisplayShelfListObj.setFormat("./photos400/photo%02d.jpg");
        //Add the object to the DisplayList hierarchy.
        b_DisplayShelfList_root.addChild(DisplayShelfListObj.obj);
}
```

6 Bearbeiten Sie die generierten HTML-Dateien. In dem Teil der HTML-Seiten, der den Text „Description text goes here“ enthält, ersetzen Sie den Text durch den HTML-Code, den Sie verwenden möchten, um von der HTML-Seite auf die Anwendung zuzugreifen. Mit dem folgenden Code werden Schaltflächen zur Steuerung der Bild-Beispielanwendung hinzugefügt:

```
<h2>Test controls</h2>
<ul>
<li><input type="button" onclick="DisplayShelfListObj.setCurrentPosition(0)"
value="Go to first item"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setCurrentPosition(3)"
value="Go to fourth item"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setSelectedIndex(0)"
value="Go to first item (with animation)"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setSelectedIndex(3)"
value="Go to fourth item (with animation)"/>
</li>
<li><input type="button" onclick="alert(DisplayShelfListObj.getAngle())"
value="Get photo angle"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setAngle(15);"
value="Set photo angle to 15&deg;"/>
</li>
</ul>
```

7 Öffnen Sie die HTML-Seite in einem Webbrowser, um die Anwendung auszuführen.

Kapitel 9: Verwenden von Flash Builder mit Flash Professional

Mit Flash Professional-Projekten können Sie auf Code in FLA- oder XFL-Dateien, die mit Flash Professional CS5.5 erstellt wurden, zugreifen. In der Regel erstellen Sie ein Projekt und Dateien in Flash Professional, und anschließend erstellen Sie ein entsprechendes Projekt in Flash Builder zum Bearbeiten und Debuggen der Dateien. Damit können Flash Professional-Entwickler die in Flash Builder verfügbare Bearbeitungs- und Debug-Umgebung nutzen.

Hinweis: *Die Funktionen von Flash Professional-Projekten sind in Flash Builder nur verfügbar, wenn Flash Professional CS5.5 installiert ist.*

Flash Professional-Projekt erstellen

Zum Erstellen eines Flash Professional-Projekts in Flash Builder müssen Sie bereits das Projekt in Flash Professional gestartet haben. Das heißt, die FLA- oder XFL-Datei muss vorhanden sein.

1. Wählen Sie „Datei“ > „Neues Flash Professional-Projekt“.
2. Wechseln Sie zur FLA- oder XFL-Zieldatei für das Projekt.

   Der Name der Datei wird der Name des Projekts.
3. Geben Sie den Projektpfad ein.

   Sie können entweder den Standardprojektpfad im Arbeitsbereich verwenden oder zu einem neuen Projektpfad navigieren.
4. Klicken Sie auf „Fertig stellen“.

   Flash Builder öffnet das neue Projekt im Paket-Explorer. Auf den Ordner mit der FLA-Zieldatei kann zugegriffen werden. Die ausgewählte FLA-Datei wird die Zieldatei des Projekts. ActionScript-Dateien, die von den Zieldateien abhängen, sind für die Bearbeitung verfügbar.

   Wird Flash Professional noch nicht ausgeführt, so wird es gestartet.

Was man in einem Flash Professional-Projekt machen kann

Für das Arbeiten mit Quelldateien eines Flash Professional-Projekts stehen folgende Möglichkeiten zur Verfügung:

- Bearbeiten der ActionScript-Dateien, die von der FLA-Zieldatei abhängen
- Debuggen der Datei im Flash Builder-Debugger oder im Flash Professional-Debugger.

  Beim Bearbeiten der Projektdateien in Flash Builder können Sie in den ActionScript-Dateien Haltepunkte setzen.

  Zum Debuggen in Flash Builder wählen Sie „Ausführen“ > „*Datei* debuggen“ oder klicken Sie in der Symbolleiste auf die Schaltfläche zum Debuggen.

Um die Datei in Flash Professional zu debuggen, wählen Sie „Ausführen“ > „Film debuggen“ oder klicken Sie in Flash Builder auf die Schaltfläche „Film debuggen“. Der Flash Professional-Debugger erkennt Haltepunkte, die mit Flash Builder gesetzt wurden.

- Veröffentlichen der Datei in Flash Professional CS5.5:

  Wählen Sie „Projekt“ > „Film veröffentlichen“ oder klicken Sie in der Symbolleiste auf die Schaltfläche „In Flash Professional veröffentlichen“.

- Ausführen der Datei in Flash Builder oder in Flash Professional:

  Zum Ausführen der Datei in Flash Builder wählen Sie „Ausführen“ > „*Datei* ausführen“ oder klicken Sie in der Symbolleiste auf die Schaltfläche zum Ausführen.

  Zum Ausführen der Datei in Flash Professional wählen Sie „Ausführen“ > „Film testen“ oder klicken Sie in der Symbolleiste auf die Schaltfläche „Film testen“.

Projekteigenschaften für Flash Professional-Projekte festlegen

1. Wählen Sie „Projekt“ > „Eigenschaften“ > „Flash Professional“.
2. Wählen Sie „Hinzufügen“, um dem Projekt weitere Dateien hinzuzufügen.

   Ein Projekt kann nicht mehr als eine FLA- oder XFL-Zieldatei als Standardzieldatei haben. Geben Sie die Standardzieldatei des Projekts über die Schaltfläche „Als Standard festlegen“ an.
3. Klicken Sie auf „OK“.

Flash-Komponenten erstellen und bearbeiten

Adobe Flash® Professional CS 5.5 erstellt Anwendungen, die mit Adobe Flash Player 10 oder höher kompatibel sind. Mit Flex erstellte Anwendungen unterstützen auch Flash Player 10 oder höher. Das bedeutet, dass Sie Elemente aus Flash Professional CS 5.5 importieren und in Anwendungen verwenden können. Sie können Steuerelemente, Container, Skins und andere Elemente in Flash Professional CS5.5 erstellen und diese Elemente dann als SWC-Dateien in die Anwendung importieren. Informationen zum Erstellen von Komponenten mit Flash Professional CS5.5 finden Sie unter Flex Skin Design-Erweiterungen und Flex Component Kit for Flash Professional.

In der Designansicht des Flash Builder-Editors können Sie eine neue Flash-Komponente einfügen, indem Sie einen Platzhalter für eine Flash-Komponente oder einen Container hinzufügen. Sie können Flash Professional CS5.5 von Flash Builder aus aufrufen, um die Komponente oder den Container zu erstellen. Sie können Flash Professional CS5.5 auch aufrufen, um zu einem früheren Zeitpunkt erstellte Flash-Komponenten zu bearbeiten.

Wenn die Anwendung eine SWFLoader-Komponente zum Starten der SWF-Dateien enthält, können Sie Flash Professional CS5.5 aufrufen, um die dazugehörigen FLA- und SWF-Dateien zu erstellen bzw. zu bearbeiten.

Flash-Komponente oder Flash-Container erstellen

1. Stellen Sie im Designmodus des Flash Builder-Editors sicher, dass die Komponentenansicht angezeigt wird.
2. Ziehen Sie in der Komponentenansicht aus dem Ordner „Benutzerdefiniert“ entweder eine neue Flash-Komponente oder einen neuen Flash-Container in den Designbereich.

Sie können die Komponente oder den Container skalieren und/oder an der gewünschten Stelle platzieren.

3 Wählen Sie im Kontextmenü oder der Standardansicht des Fensters „Dateieigenschaften“ die Option „In Adobe Flash erstellen“ aus.

Hinweis: *Sie können auch in der Designansicht auf die Komponente doppelklicken, um das Element in Adobe Flash zu erstellen.*

4 Geben Sie im Dialogfeld, das jetzt angezeigt wird, Namen für die Klasse und die SWC-Datei ein. Klicken Sie anschließend auf „Erstellen“, um Adobe Flash Professional CS5.5 zu öffnen.

5 Bearbeiten Sie die Komponente oder den Container in Flash Professional CS5.5. Klicken Sie anschließend auf „Fertig“, um wieder zu Flash Builder zurückzukehren.

Flash-Komponente oder Flash-Container bearbeiten

Bei den folgenden Schritten wird angenommen, dass Sie bereits eine Flash-Komponente oder einen Flash-Container in Flash Builder eingefügt haben.

1 Wählen Sie im Designmodus des Flash Builder-Editors die Flash-Komponente oder den Flash-Container aus, die bzw. den Sie bearbeiten möchten.

2 Wählen Sie im Kontextmenü oder der Standardansicht des Fensters „Flex-Eigenschaften“ die Option „In Adobe Flash bearbeiten“ aus.

Hinweis: *Sie können auch in der Designansicht auf die Komponente doppelklicken, um das Element in Adobe Flash zu bearbeiten.*

3 Bearbeiten Sie die Komponente oder den Container in Flash Professional CS5.5. Klicken Sie anschließend auf „Fertig“, um wieder zu Flash Builder zurückzukehren.

Mit SWFLoader-Komponente verknüpfte SWF-Dateien erstellen oder bearbeiten

Bei den folgenden Schritten wird angenommen, dass die Anwendung eine SWFLoader-Komponente enthält.

1 Wählen Sie im Designmodus des Flash Builder-Editors die SWFLoader-Komponente aus.

2 Starten Sie Flash Professional CS5.5 über das Kontextmenü für die Komponente oder die Standardansicht des Fensters „Flex-Eigenschaften“. Führen Sie hierzu einen der folgenden Schritte aus:

- Wählen Sie „In Adobe Flash erstellen“ aus, um eine SWF-Datei zu erstellen, die mit der SWFLoader-Komponente verknüpft ist.
- Wählen Sie „In Adobe Flash bearbeiten“ aus, um die SWF-Datei zu bearbeiten, die mit der SWFLoader-Komponente verknüpft ist.

3 Klicken Sie anschließend auf „Fertig“, um wieder zu Flash Builder zurückzukehren.

Elemente aus Flash CS3 oder höher importieren

Sie können Flash-Komponenten zur Benutzeroberfläche hinzufügen, die in Adobe Flash CS3 Professional erstellt wurden.

Hinweis: *Bevor Sie Flex-Komponenten in Flash CS3 erstellen können, müssen Sie das Flex Component Kit für Flash installieren. Weitere Informationen hierzu finden Sie im Artikel Importieren von Adobe Flash CS Professional-Elementen in Adobe Flex.*

1 Speichern Sie die Flash-Komponente im Bibliothekspfad des aktuellen Projekts, sofern dies noch nicht geschehen ist.

Der Bibliothekspfad gibt den Pfad einer oder mehrerer SWC-Dateien an, zu dem die Anwendung bei der Kompilierung einen Link herstellt. Der Pfad ist in den Flex-Compiler-Einstellungen für das Projekt definiert. Bei neuen Projekten befindet sich der Ordner `libs` standardmäßig im Bibliothekspfad.

Um den Bibliothekspfad festzulegen oder zu ermitteln, wählen Sie das Projekt im Paket-Explorer aus und klicken Sie dann auf „Projekt“ > „Eigenschaften“. Wählen Sie im Dialogfeld „Eigenschaften“ die Kategorie „Flex-Erstellungspfad“ aus und klicken Sie dann auf die Registerkarte „Bibliothekspfad“. Weitere Informationen hierzu finden Sie unter „Projekte manuell erstellen“ auf Seite 111.

2 Öffnen Sie eine MXML-Datei und fügen Sie auf eine der folgenden Weisen eine Flash-Komponente hinzu:

- Erweitern Sie im Designmodus des MXML-Editors in der Komponentenansicht die Kategorie „Benutzerdefiniert“ und ziehen Sie die Flash-Komponente in die MXML-Datei. Klicken Sie bei bereits geöffneten Dokumenten auf die Schaltfläche „Aktualisieren“ (das Symbol mit den beiden kreisförmig angeordneten grünen Pfeilen), um die Komponente anzuzeigen, nachdem Sie sie eingefügt haben.
- Geben Sie im Quellmodus das Tag für die Komponente ein und verwenden Sie dann die Inhaltshilfe, um das Tag schnell zu vervollständigen.

Flash Player-Sicherheit verwalten

In Flash Player kann eine Anwendung nur mit expliziter Erlaubnis Daten empfangen, die aus einer anderen als der Domäne stammen, in der sie geladen wurde. Wenn Sie die SWF-Anwendungsdatei in http://mydomain.com geladen haben, kann sie keine Daten aus http://yourdomain.com laden. Diese Sicherheitssandbox verhindert, dass Flash Player-Funktionen in böswilliger Absicht verwendet werden. (JavaScript verfügt über ein ähnliches Sicherheitsmodell zum gleichen Zweck.)

Sie haben drei Möglichkeiten, von einer in Flex erstellten Anwendung aus auf Daten zuzugreifen:

- Fügen Sie eine Cross-Domain-Richtliniendatei im Stammordner des Webservers hinzu, der den Datendienst hostet.
- Legen Sie die SWF-Anwendungsdatei auf dem Server ab, der den Datendienst hostet.
- Erstellen Sie auf dem gleichen Server, der auch die SWF-Anwendungsdatei enthält, einen Proxy, der einen auf einem anderen Server gehosteten Datendienst aufruft.

Cross-Domain-Richtliniendateien verwenden

Eine Cross-Domain-Richtliniendatei ist eine einfache XML-Datei, die Flash Player den Zugriff auf Daten in einer anderen Domäne ermöglicht. Ohne diese Richtliniendatei wird der Benutzer aufgefordert, über ein Dialogfeld die Zugriffsberechtigung zu gewähren. Eine derartige Situation sollte vermieden werden.

Die Cross-Domain-Richtliniendatei (crossdomain.xml) wird im Stammordner des bzw. der Server platziert, auf dem sich die gewünschten Daten befinden. Beispiel für eine Cross-Domain-Richtliniendatei:

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM "http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
    <allow-access-from domain="www.yourdomain.com" />
</cross-domain-policy>
```

Weitere Informationen zum Konfigurieren von Cross-Domain-Richtliniendateien finden Sie unter Sicherheit.

Proxys für den Zugriff auf Remotedaten einrichten

Eine andere Möglichkeit zur Verwaltung der Flash Player-Sicherheit (neben einer Cross-Domain-Richtliniendatei) ist die Verwendung eines Proxys. LiveCycle Data Services und BlazeDS bieten ein komplettes Proxyverwaltungssystem für in Flex erstellte Anwendungen. Sie können auch einen einfachen Proxydienst mit einer Web-Scripting-Sprache wie ColdFusion, JSP, PHP oder ASP erstellen.

Der Proxydienst verarbeitet Anforderungen, die von der Anwendung an den Remotedienst gesendet werden, und die Antworten des Remotediensts an die Anwendung (Flash Player).

Eine gängige Technik bei der Anwendungsentwicklung ist das Hosten des Proxys auf dem lokalen Computer. Dazu müssen ein Webserver und eine Scripting-Sprache auf dem lokalen Entwicklungscomputer ausgeführt werden.

Flash Builder zur Verwendung eines Proxys für den Zugriff auf Remotedaten einrichten

Nachdem Sie einen Proxy für den Zugriff auf Daten eines Remotediensts eingerichtet haben, legen Sie die Anwendungsdateien in derselben Domäne wie derjenigen des Proxys ab. In Flash Builder können Sie die Projekterstellungseinstellungen und die Startkonfiguration zur Verwendung eines Proxys ändern.

Wenn Sie Anwendungen in Flash Builder kompilieren und der Proxyserver ebenfalls auf dem lokalen Entwicklungscomputer eingerichtet ist, können Sie die Projekterstellungseinstellungen ändern, um die kompilierten Anwendungsdateien automatisch an den gewünschten Speicherort auf dem Webserver zu kopieren.

Projekterstellungspfads ändern

1. Wählen Sie im Flex Paket-Explorer ein Projekt aus.
2. Klicken Sie mit der rechten Maustaste und wählen Sie „Eigenschaften“. Das Dialogfeld mit den Projekteigenschaften wird angezeigt.
3. Wählen Sie die Eigenschaftenseite „Flex-Erstellungspfad“ aus.
4. Ändern Sie den vorhandenen Ausgabeordner, indem Sie einen neuen Pfad eingeben, oder indem Sie zum gewünschten Ordner auf dem Webserver navigieren (z. B. C:\inetpub\wwwroot\meineAnwendung\).
5. Klicken Sie auf „OK“.

Um die Anwendung über den Webserver auszuführen und zu debuggen, ändern Sie die Startkonfiguration des Projekts.

Hinweis: *Wenn Ihr Proxyserver nicht Ihr lokaler Computer ist, kopieren Sie den Inhalt des Ordners „bin“ auf Ihren Server, bevor Sie Ihr Programm ausführen oder debuggen.*

Startkonfiguration ändern

1. Öffnen Sie die Hauptanwendungsdatei des Projekts in Flash Builder, klicken Sie mit der rechten Maustaste in den Editor und wählen Sie „Ausführen als“ > „Konfigurationen ausführen“. Das Dialogfeld „Konfigurationen erstellen, verwalten und ausführen“ wird angezeigt.

2 Wählen Sie aus der Liste der Konfigurationen die Startkonfiguration des Projekts.

3 (Optional) Deaktivieren Sie auf der Registerkarte „Basis“ die Option „Standardeinstellung wiederherstellen“, um die Anfangs-URL bzw. den Anfangspfad zu ändern.

4 Geben Sie im Textfeld „Debug“ die URL oder den Pfad zur Debugversion der Anwendung ein.

5 Geben Sie im Textfeld „Profil“ die URL oder den Pfad zur Profiler-Version der Anwendung ein.

6 Geben Sie im Textfeld „Ausführen“ die URL oder den Pfad zur Hauptanwendungsdatei ein.

7 Klicken Sie auf „Anwenden“ und dann auf „Schließen“.

Kapitel 10: Verwenden von Flash Builder mit Flash Catalyst

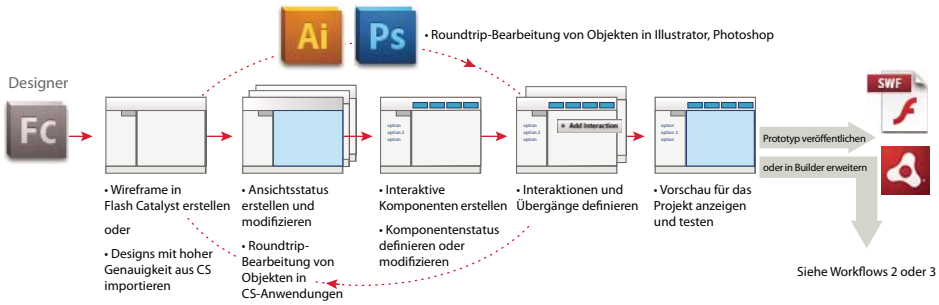
Arbeitsabläufe zwischen Flash Catalyst und Flash Builder

In Flash Catalyst CS5.5 und Flash Builder 4.5 oder höheren Versionen können Designer und Entwickler gemeinsam an Projekten arbeiten. Es folgt eine Übersicht über drei gängige Arbeitsabläufe.

Dieses Video vom Adobe Community-Experten Steven Peeters gibt einen Überblick über die Arbeitsabläufe zwischen Flash Catalyst und Flash Builder.

Wireframe- und Prototyparbeitsablauf in Flash Catalyst

In diesem Arbeitsablauf werden allgemeine Schritte beschrieben, die Designer oft in Flash Catalyst und anderen CS-Anwendungen ausführen werden, um Wireframe und schnell einen Prototyp von Anwendungen zu erstellen. Wenn ein Entwickler beteiligt ist, können Flash Catalyst-Projekte mithilfe von Flash Builder erweitert werden – beispielsweise zum Herstellen einer Verbindung zu einer Datenbank oder zu Webdiensten. Die Arbeitsabläufe 2 und 3 veranschaulichen, wie Dateien und Anwendungsbestandteile (z. B. Skins und Komponenten) zwischen Designern und Entwicklern ausgetauscht werden können.



Anwendung planen Beginnen Sie mit einer detaillierten Projektspezifikation. Darin wird jede Seite bzw. jeder Bildschirm beschrieben, einschließlich Bildmaterial und interaktiven Komponenten, Benutzernavigation sowie Status der Komponenten. Die Spezifikation beschreibt weiterhin die gegebenenfalls verwendeten Datenlistenkomponenten, mit denen externe Daten abgerufen und angezeigt werden.

Wireframe erstellen oder ein voll vertrauenswürdiges Design importieren Erstellen Sie mithilfe der grafischen Werkzeuge in Flash Catalyst ein Wireframe des Anwendungslayouts oder importieren Sie ein Design aus Adobe Illustrator, Photoshop oder Fireworks.

Bildmaterial, Video und Sound erstellen bzw. beschaffen Erstellen Sie für die Anwendung zusätzliches Bildmaterial und zusätzliche Videos und Sounds.

Bildmaterial, Video und Sound integrieren Fügen Sie aus mehreren Schichten bestehendes Bildmaterial in Flash Catalyst ein. Sie können auch einzelne Grafikdateien importieren oder mithilfe der integrierten Vektorzeichenwerkzeuge einfache Grafiken erstellen. Importieren Sie zusätzliche Elemente, wie beispielsweise Videos, Sounds und SWF-Inhalte. Importieren Sie für datenorientierte Komponenten, wie beispielsweise eine Datenliste, eine repräsentative Datenprobe (Text oder Bilder). Weitere Informationen finden Sie unter Importieren von Bildmaterial.

In Flash Catalyst importiertes oder erstelltes Bildmaterial können Sie in Illustrator oder Photoshop starten und bearbeiten und anschließend wieder in Flash Catalyst übernehmen. Die integrierte Bearbeitung erweitert die Zeichen- und Grafikbearbeitungsfunktionen von Flash Catalyst und verbessert den iterativen Designprozess. Weitere Informationen finden Sie unter Integrierte Bearbeitung.

Ansichtsstatus erstellen und modifizieren Erstellen Sie die Status entsprechend den Vorgaben der Projektspezifikation. Weitere Informationen finden Sie unter Statustypen.

Interaktive Komponenten erstellen und Komponentenstatus definieren Konvertieren Sie Bildmaterial in vorgefertigte Komponenten (Schaltflächen, Bildlaufleisten, Datenlisten usw.). Häufig verwendete Komponenten mit allgemeinem Aussehen können Sie über das Bedienfeld „Allgemeine Bibliothek“ schnell hinzufügen. Erstellen Sie für Verhalten, das nicht durch integrierte Komponenten bereitgestellt wird, benutzerdefinierte Komponenten. Weitere Informationen finden Sie unter Was ist eine Komponente?

Verwenden Sie für das Design von Datenlistenkomponenten in datenorientierten Anwendungen Entwurfszeitdaten. Als Entwurfszeitdaten können Sie simulierte Inhalte verwenden, z. B. musterhafte Datenbankdatensätze oder Bilder, ohne hierfür eine Verbindung zum Backend-System herstellen zu müssen. Der Flex-Entwickler ersetzt später die Entwurfszeitdaten durch echte Daten aus einer Datenbank oder einem Webservice. Weitere Informationen zu Entwurfszeitdaten finden Sie unter Datenlisten und Bedienfelder mit Bildlauf.

Komponentenstatus erstellen oder modifizieren Komponenten können mehrere Status aufweisen. Eine Schaltfläche hat z. B. die Status „Normal“, „Darüber“, „Gedrückt“ und „Deaktiviert“. Erstellen bzw. modifizieren Sie die verschiedenen Status der interaktiven Komponenten gemäß Projektspezifikation.

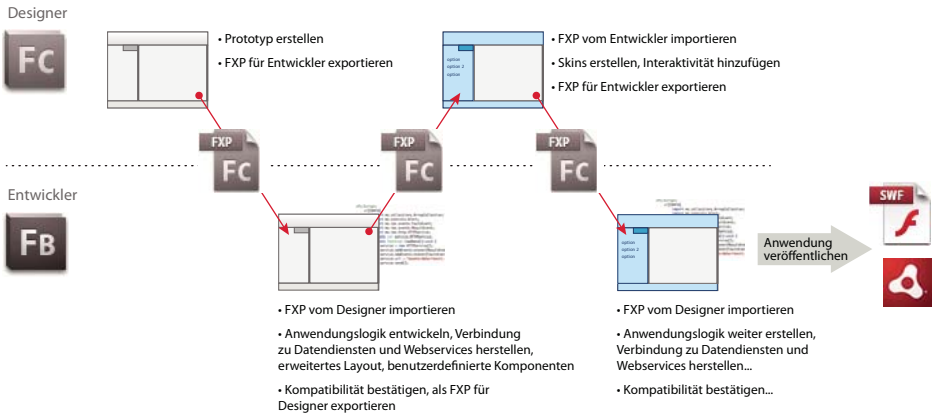
Hinweis: *Die Arbeitsschritte für die Erstellung von Seitenstatus und interaktiven Komponenten sind austauschbar. Einige Designer bevorzugen es, zuerst die interaktiven Komponenten zu erstellen und diese anschließend den Seiten und Status hinzuzufügen.*

Interaktionen und Übergänge definieren Fügen Sie Interaktionen hinzu, die definieren, was bei einer Interaktion des Benutzers mit der Anwendung geschehen soll. Sie können z. B. festlegen, dass nach dem Klicken des Benutzers auf eine Schaltfläche ein Seiten- bzw. Komponentenstatus in einen anderen Status übergeht. Sie können auch Interaktionen hinzufügen, durch die eine Animation abgespielt, eine Videowiedergabe gesteuert oder eine URL geöffnet wird. Im Bedienfeld „Zeitachsen“ können Sie glatte, animierte Übergänge zwischen Seiten- und Komponentenstatus hinzufügen und modifizieren. Weitere Informationen zu Interaktionen finden Sie unter Interaktionen. Weitere Informationen zu Übergängen finden Sie unter Übergänge und Aktionssequenzen.

Projekt testen Testen Sie das Projekt während der Entwicklung häufig, um sicherzustellen, dass die Interaktionen ordnungsgemäß ausgeführt werden. Siehe Veröffentlichen.

Arbeitsablauf für ein kleines Team zwischen Flash Catalyst und Flash Builder

Für einfachere Projekte mit einem Team aus ein bis zwei Personen kann mithilfe von FXP-Dateien ein komplettes Flex-Projekt zwischen einem Designer, der mit Flash Catalyst arbeitet, und einem Entwickler, der mit Flash Builder arbeitet, ausgetauscht werden. Das FXP-Format ist ein Archivformat, das Projektordner, Dateien und Metadaten zum Projekt enthält.


Designer
• Prototyp erstellen
• FXP für Entwickler exportieren
• FXP vom Entwickler importieren
• Skins erstellen, Interaktivität hinzufügen
• FXP für Entwickler exportieren
FXP
FXP
FXP
Entwickler
• FXP vom Designer importieren
• Anwendungslogik entwickeln, Verbindung zu Datendiensten und Webservices herstellen, erweitertes Layout, benutzerdefinierte Komponenten
• Kompatibilität bestätigen, als FXP für Designer exportieren
• FXP vom Designer importieren
• Anwendungslogik weiter erstellen, Verbindung zu Datendiensten und Webservices herstellen...
• Kompatibilität bestätigen...
Anwendung veröffentlichen
SWF

Eine FXP-Datei enthält alle erforderlichen Elemente, um mit dem Projekt in Flash Builder zu arbeiten. Der Entwickler kann dem Designer mithilfe der FXP-Datei Skins, ein Basislayout, Animationen und einfache Interaktivität für das Projekt bereitstellen.

Mithilfe von Flash Builder kann der Entwickler Unternehmenslogik, Datenkonnektivität, ein erweitertes Layout und benutzerdefinierte skinfähige Komponenten hinzufügen. Der Entwickler kann die FXP-Datei an den Designer zurücksenden, damit er im Laufe des Projekts die Skins weiter bearbeiten sowie Interaktivität und Animationen hinzufügen kann. Dabei muss der Entwickler auf die Einhaltung der Kompatibilität mit Flash Catalyst im Projekt achten. Weitere Informationen zur Kompatibilität mit Flash Catalyst in Flash Builder finden Sie unter Richtlinien zum Erstellen eines mit Flash Catalyst kompatiblen Projekts.

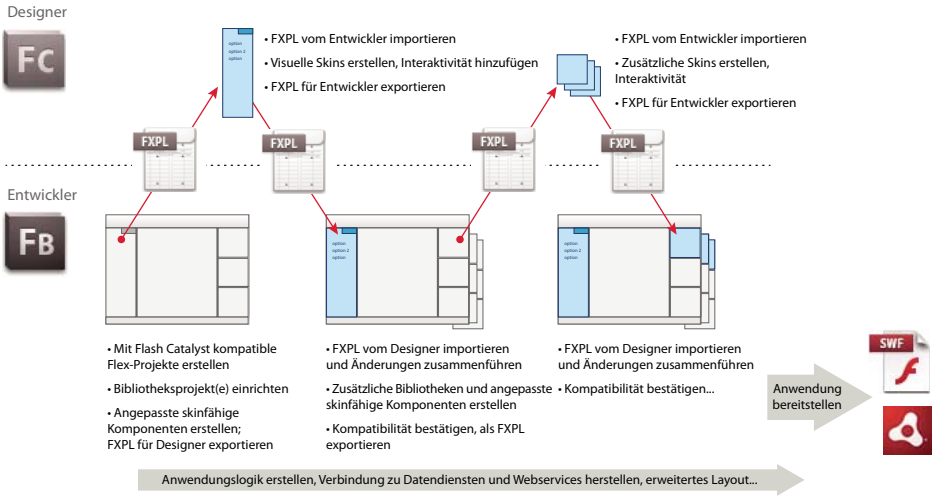
Falls zusätzliche Designschritte erforderlich sind, kann der Designer die FXP-Datei in Flash Catalyst erneut öffnen und die Skins, das Basislayout und die grundlegende Interaktivität des Projekts bearbeiten. Anschließend kann der Designer die FXP-Datei erneut speichern und dem Entwickler das Übernehmen dieser Änderungen für das Projekt in Flash Builder erlauben.

Hinweis: *Einige Komponenten und sonstige Aspekte des Projekts können möglicherweise in Flash Catalyst nicht bearbeitet werden, nachdem sie in Flash Builder geändert wurden.*

Am Ende dieses Vorgangs kann das Projekt in Flash Builder abgeschlossen und veröffentlicht werden.

Arbeitsablauf mit mehreren Personen zwischen Flash Builder und Flash Catalyst

Für komplexere Projekte können die Entwickler Bearbeitungen der Designer an Teilen des Designs in Flash Catalyst abwarten und dann lediglich diese Teile in ein Flex-Bibliotheksprojekt verschieben. Anschließend kann er das Bibliotheksprojekt vom Hauptprojekt aus referenzieren und das Bibliotheksprojekt als FXPL-Datei exportieren. Der Designer kann die FXPL-Datei in Flash Catalyst importieren, die nötigen Änderungen vornehmen und die FXPL-Datei exportieren, um sie an den Entwickler zurückzusenden.


Designer
Entwickler
• FXPL vom Entwickler importieren
• Visuelle Skins erstellen, Interaktivität hinzufügen
• FXPL für Entwickler exportieren
• FXPL vom Entwickler importieren
• Zusätzliche Skins erstellen, Interaktivität
• FXPL für Entwickler exportieren
FXPL
FXPL
FXPL
FXPL
• Mit Flash Catalyst kompatible Flex-Projekte erstellen
• Bibliotheksprojekt(e) einrichten
• Angepasste skinfähige Komponenten erstellen; FXPL für Designer exportieren
• FXPL vom Designer importieren und Änderungen zusammenführen
• Zusätzliche Bibliotheken und angepasste skinfähige Komponenten erstellen
• Kompatibilität bestätigen, als FXPL exportieren
• FXPL vom Designer importieren und Änderungen zusammenführen
• Kompatibilität bestätigen...
Anwendung bereitstellen
SWF
Anwendungslogik erstellen, Verbindung zu Datendiensten und Webservices herstellen, erweitertes Layout...

Das Projekt ist in ein Hauptprojekt und untergeordnete Komponentenbibliotheken aufgeteilt. Nur die Bibliotheken werden in Flash Catalyst bearbeitet. Informationen zum Erstellen einer Bibliothek in Flash Builder finden Sie unter „Flex-Bibliotheksprojekte erstellen“ auf Seite 235. Weitere Informationen zum Arbeiten mit Bibliotheken in Flash Catalyst finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 234.

Eine FXPL-Datei wird aus der Projektbibliothek exportiert und enthält Informationen zur Komponente, die ein Entwickler zum Arbeiten mit dieser Komponente in Flash Builder benötigt. In Flash Builder integriert der Entwickler die FXPL-Datei im Rahmen eines größeren Projekts.

Der Entwickler kann FXPL-Dateien als benutzerdefinierte skinfähige Komponenten, die in Flash Catalyst bearbeitet werden können, wieder an den Designer exportieren. Weitere Informationen zum Erstellen benutzerdefinierter skinfähiger Komponenten in Flash Builder finden Sie unter „Skinfähige ActionScript-Komponente erstellen“ auf Seite 44.

Anschließend können Sie die Designelemente als neues Projekt in Flash Builder importieren und die Änderungen mithilfe der Werkzeuge von Flash Builder vergleichen und zusammenführen. Weitere Informationen finden Sie unter „Dateien zwischen Flash Builder und Flash Catalyst übergeben“ auf Seite 263

Falls zusätzliche Designschritte erforderlich sind, können Sie die FXPL-Dateien zwischen Flash Builder und Flash Catalyst übertragen. Dabei muss auf die Einhaltung der Kompatibilität mit Flash Catalyst im Projekt geachtet werden. Weitere Informationen zum Einhalten der Kompatibilität mit Flash Catalyst im Projekt finden Sie unter „Kompatibilität mit Flash Catalyst sicherstellen“ auf Seite 262.

Mithilfe der Werkzeuge in Flash Builder können Sie dann eine voll funktionsfähige Anwendung erstellen. Sie können die Datendienste definieren, die zur Laufzeit die Daten abrufen sollen, und die zurückgegebenen Daten dann an visuelle Komponenten im Projekt binden. So könnte z. B. durch Klicken auf eine Schaltfläche eine HTTPService-Komponente aufgerufen werden. Die Komponente kann Remoteprozeduraufrufe für die Interaktion mit Serverumgebungen wie ColdFusion oder PHP nutzen und Daten für die Anwendung bereitstellen. Weitere Informationen hierzu finden Sie unter Building data-centric applications with Flash Builder.

Integration von Flash Catalyst und Flash Builder

Projektstruktur zum Sicherstellen der Kompatibilität mit Flash Catalyst

Flash Catalyst CS5.5 unterstützt einen Teil der in Flex 4.5 oder höher verfügbaren Funktionen und Komponenten. Deshalb müssen Sie unbedingt das Projekt so strukturieren, dass es vollständig mit Flash Catalyst kompatibel ist.

Projekt planen

Identifizieren Sie beim Planen des Projekts die visuellen Bestandteile des Projekts, die einem Designer gehören, sowie die logischen Bestandteile, die einem Entwickler gehören. In der Regel gehören dem Designer die visuellen Objekte, Animationen und das Basislayout des Projekts. Dem Entwickler gehören in der Regel die Architektur, Funktionalität und das Layout auf Anwendungsebene des Projekts.

In Flash Catalyst und Flash Builder gibt es mehrere Werkzeuge, mit denen Sie Design und Entwicklung genau abgrenzen können. Durch die Aufteilung des Projekts in ein Hauptprojekt und mit Flash Catalyst kompatible Bibliotheksprojekte wird eindeutig definiert, welche Bestandteile des Projekts der Designer bearbeiten kann. Die Verwendung skinfähiger Komponenten ermöglicht eine klare Trennung zwischen logischen und visuellen Bestandteilen auf der Komponentenebene.

Bibliotheksprojekte verwenden

Mithilfe von Bibliotheksprojekten können Sie Design und Logik auf der Projektebene voneinander trennen. Entwickler können mithilfe von Bibliotheksprojekten Projektbestandteile, die in Flash Catalyst bearbeitet werden müssen, voneinander trennen.

Bibliotheksprojekte vereinfachen außerdem das Zusammenführen, da der Entwickler das Bibliotheksprojekt gewöhnlich nicht bearbeitet, während es vom Designer bearbeitet wird.

Ein Hauptprojekt und ein untergeordnetes Flash Catalyst-kompatibles Bibliotheksprojekt erstellen

1. Erstellen Sie das Flex-Hauptprojekt. Weitere Informationen finden Sie unter „Flex-Projekte“ auf Seite 90.
2. Erstellen Sie ein Flex-Bibliotheksprojekt.

   Stellen Sie sicher, dass das Projekt mit Flash Catalyst kompatibel ist, indem Sie im Assistenten für neue Flex-Bibliotheksprojekte die Option „Projekt mit Flash Catalyst kompatibel machen“ auswählen. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte erstellen“ auf Seite 235.
3. Fügen Sie das Bibliotheksprojekt dem Erstellungspfad des Flex-Hauptprojekts hinzu.

   Wechseln Sie dazu zu „Projekt“ > Eigenschaften“ und wählen Sie „Flex-Erstellungspfad“ aus. Klicken sie auf der Registerkarte „Bibliothek“ auf „Projekt hinzufügen“. Weitere Informationen finden Sie unter „Projekterstellungspfad ändern“ auf Seite 110.
4. Platzieren Sie die skinfähigen Komponenten und die Skins im Bibliotheksprojekt sowie die Codedateien im Hauptprojekt.

Hinweis: *Das Hauptprojekt hängt vom Bibliotheksprojekt ab. Deshalb kann die Bibliothek keinen Code enthalten, der vom Hauptprojekt abhängt, und muss eigenständig sein.*

Verwandte Themen

„Dateien zwischen Flash Builder und Flash Catalyst übergeben“ auf Seite 263

Skinfähige Komponenten verwenden

Die Skinning-Architektur in Flex ermöglicht die Erstellung erweiterbarer skinfähiger Komponenten, die mit Flash Catalyst kompatibel sind. Durch das Erstellen skinfähiger Komponenten in Flash Builder und das visuelle Erstellen von Skins in Flash Catalyst können Sie Design und Logik auf der Komponentenebene voneinander trennen.

Eine skinfähige Komponente enthält die Logik einer Komponente, während die Skin die sichtbaren Elemente und die Layoutregeln enthält. Darüber hinaus kann eine skinfähige Komponente anzeigen, dass ihre Skin Teile und Status enthält. Die Hostkomponente kann die Teile und Status der Skin programmgesteuert steuern. Durch die Einschränkung der Kommunikation zwischen der skinfähigen Komponente und deren Skins auf die Teile und Status sind die beiden Teile sehr unabhängig und flexibel. Weitere Informationen zum Erstellen benutzerdefinierter skinfähiger Komponenten in Flash Builder finden Sie unter Beispiel: Skinfähige Spark-Komponente erstellen.

Nachdem Sie eine skinfähige Komponentendefinition in Flash Builder erstellt haben, erstellen Sie eine Beispiel-Skin. Wenn Sie dabei das Projekt in Flash Catalyst importieren, wird die Skin in Flash Catalyst im Bedienfeld „Komponenten“ angezeigt. Die Skin kann dann problemlos in Flash Catalyst bearbeitet werden. Das Erstellen einer Beispiel-Skin hilft dem Designer beim Verständnis der grundlegenden Skin-Struktur.

Verwandte Themen

Benutzerdefinierte skinfähige Komponenten

„Skinfähige ActionScript-Komponente erstellen“ auf Seite 44

Skinerstellung für benutzerdefinierte Komponenten mithilfe von Flash Builder und Flash Catalyst (Video)

Kompatibilität mit Flash Catalyst sicherstellen

Wie jedes visuelle Kodierwerkzeug unterstützt auch Flash Catalyst nur einen Teil des Flex-Codes. In Flash Builder gibt es jedoch eine Kompatibilitätsprüfung, die Ihnen genau mitteilt, welche Codeelemente mit Flash Catalyst bearbeitet werden können.

Wenn Sie in Flash Builder ein mit Flash Catalyst kompatibles Projekt erstellen, wird die Flash Catalyst-Kompatibilitätsprüfung („Projekt“ > „Eigenschaften“ > „Flash Catalyst“) automatisch aktiviert. Wenn Sie ein Projekt in Flash Catalyst erstellen und in Flash Builder importieren, wird es automatisch so konfiguriert, dass es mit Flash Catalyst kompatibel ist. Wenn Ihre Änderungen am Projekt jedoch Kompatibilitätsprobleme verursachen, zeigt Flash Builder die Kompatibilitätsfehler in der Problemansicht an und weist darauf hin, welche Codeelemente in Flash Catalyst bearbeitet werden können.

Sie müssen nicht notwendigerweise alle Kompatibilitätsprobleme beheben, bevor Sie ein Projekt in Flash Catalyst öffnen. In der Spalte "Typ" der Problemansicht werden die Auswirkungen der einzelnen Kompatibilitätsprobleme angezeigt. Probleme, die das Öffnen des Projekts in Flash Catalyst verhindern, werden als Warnungen angezeigt. Weniger schwerwiegende Probleme werden als Benachrichtigungen vom Typ "Info" angezeigt.

Inkompatibilitätstyp	Beschreibung
Inkompatibilität mit Projekt	Das Projekt kann nicht in Flash Catalyst geöffnet werden.
Inkompatibilität mit Datei	Die Datei kann nicht in Flash Catalyst bearbeitet werden. Bei Inkompatibilitäten der Hauptanwendung ist das Projekt inkompatibel.
Inkompatibilität mit skinfähiger Komponente	Für die Komponente können in Flash Catalyst keine Skins erstellt werden, oder das Skinteil kann in Flash Catalyst nicht zugewiesen werden.
Inkompatibilität mit Entwurfszeitdaten	Die Listendaten werden durch Anwendungscode gesteuert, der in Flash Catalyst nicht bearbeitet werden kann. Sie können die Darstellung der Liste in Flash Catalyst bearbeiten.
Kompatibilitätswarnung für Flash Catalyst	Durch diese Warnung wird die Bearbeitungsmöglichkeit des Projekts in Flash Catalyst nicht eingeschränkt. Dies kann jedoch ein Hinweis darauf sein, dass beim Bearbeiten geringfügige Probleme auftreten.

Weitere Informationen zu Flash Catalyst-Kompatibilitätsproblemen finden Sie unter Richtlinien zum Erstellen eines mit Flash Catalyst kompatiblen Projekts.

Verwandte Themen

Sicherstellen der Kompatibilität in Flash Builder- und Flash Catalyst-Projekten (Video)

Dateien zwischen Flash Builder und Flash Catalyst übergeben

FXP-Dateien dienen zum Austauschen kompletter Flex-Projekte oder von Komponenten, Komponentenskins und Elementen zwischen Designern und Entwicklern.

Für komplexe Projekte können Sie mithilfe eines Bibliotheksprojekts die Benutzeroberfläche des Projekts als Abfolge skinfähiger Komponenten und Skins strukturieren. Weitere Informationen finden Sie unter „Flex-Bibliotheksprojekte“ auf Seite 234.

Anschließend können Sie FXPL-Dateien zwischen Flash Builder und Flash Catalyst übertragen, um nur die Komponentenskins und Designelemente auszutauschen. Um einen ordnungsgemäßen Ablauf zu gewährleisten, kann der Designer die FXPL-Datei in ein leeres Projekt in Flash Catalyst importieren, visuelle Skins erstellen und Interaktivität hinzufügen.

Mithilfe der folgenden Export- und Importarbeitsabläufe können Sie die FXP- und FXPL-Dateien zwischen Flash Builder und Flash Catalyst übertragen.

Dateien von Flash Builder an Flash Catalyst übergeben

- Wählen Sie zum Exportieren einer FXP- oder FXPL-Datei aus Flash Builder in Flash Catalyst „Projekt“ > „Flash Catalyst“ > „Flash Catalyst-Projekt exportieren“ aus.
- Wechseln Sie zum Importieren der FXP- oder FXPL-Datei aus Flash Builder in Flash Catalyst zu „Datei“ > „Importieren“ und wählen Sie die Adobe FXG-Datei (.fxg) oder das Bibliothekspaket (.fxpl) aus.

Dateien von Flash Catalyst an Flash Builder übergeben

- Wechseln Sie zum Exportieren einer FXP- oder FXPL-Datei aus Flash Catalyst in Flash Builder zu „Datei“ > „Exportieren“ und wählen Sie die Adobe FXG-Datei (.fxg) oder das Bibliothekspaket (.fxpl) aus.
- Wählen Sie zum Importieren einer FXP- oder FXPL-Datei aus Flash Catalyst in Flash Builder „Projekt“ > „Flash Catalyst“ > „Flash Catalyst-Projekt importieren“ aus.

FXP-Dateien mit dem Befehl „Projekt in Flash Catalyst bearbeiten“ bearbeiten

Falls sowohl Flash Builder als auch Flash Catalyst auf demselben Computer installiert sind, können Sie FXP-Dateien mit dem Befehl „Projekt in Flash Catalyst bearbeiten“ bearbeiten.

Weitere Informationen finden Sie unter „Befehl „Projekt in Flash Catalyst bearbeiten““ auf Seite 264.

Änderungen aus Flash Catalyst und Flash Builder zusammenführen

Beim Importieren von FXP- oder FXPL-Dateien aus Flash Catalyst in Flash Builder können Sie die Unterschiede zwischen den beiden Projekten mit dem Vergleichseditor vergleichen und zusammenführen. Sie können auch ein Werkzeug eines anderen Herstellers verwenden.

Beim Erstellen von Designelementen in Flash Catalyst können Sie die Elemente in Flash Builder integrieren, indem Sie die FXP- oder FXPL-Datei mithilfe der folgenden Methoden importieren.

- Importieren der Designelemente in ein neues Projekt und anschließendes Vergleichen und Zusammenführen der Änderungen mithilfe der Werkzeuge von Flash Builder
- Importieren des Inhalts der Bibliothek in das bestehende Projekt
- Nur Überschreiben der bestehenden FXP-Datei mit der aktualisierten Version, wenn in der Zwischenzeit keine Änderungen daran vorgenommen wurden

  Hinweis: *Bibliotheksprojekte (.fxpl) können nicht überschrieben werden.*

Zum Vergleichen von Projekten mithilfe des Vergleichseditors wählen Sie die Projekte aus, indem Sie die Strg-Taste drücken und auf die zu vergleichenden Projekte klicken. Klicken Sie anschließend mit der rechten Maustaste auf die ausgewählten Projekte und wählen Sie „Vergleichen mit“ > „Einander“ aus. Die Ergebnisse werden im Bedienfeld „Textvergleich“ angezeigt.

Hinweis: *Beim Zusammenführen sind mehrere Änderungen bei Einstellungsdateien (z. B. „actionScriptProperties“) oder Ausgabedateien (z. B. die Datei „bin-debug“) oder sonstigen Projektkomponenten sichtbar. In der Regel können Sie diese Änderungen ignorieren und sich ganz auf die Änderungen beim Quellordner konzentrieren.*

Weitere Informationen zur Verwendung des Vergleichseditors finden Sie in der Dokumentation zu Eclipse.

Befehl „Projekt in Flash Catalyst bearbeiten“

Verwenden Sie den Befehl „Projekt in Flash Catalyst bearbeiten“, wenn sowohl Flash Builder als auch Flash Catalyst auf demselben Computer installiert sind. Sie können Flash Catalyst von Flash Builder aus starten und Änderungen am Design direkt im Projekt vornehmen, ohne die Projektdateien zwischen Flash Builder und Flash Catalyst zu übertragen.

1. Um Flash Catalyst direkt von Flash Builder aus zu starten, wählen Sie Projekt“ > „Flash Catalyst“ > „Projekt in Flash Catalyst bearbeiten“ aus.

   Sie können auch im Kontextmenü für das Projekt „Flash Catalyst“ >„Projekt in Flash Catalyst bearbeiten“ auswählen.

2. Ändern Sie das Anwendungsdesign nach Ihren Wünschen. Wenn Sie in Flash Catalyst Änderungen am Projekt vornehmen, wird dieses in Flash Builder gesperrt. Durch Sperren des Projekts wird sichergestellt, dass es nicht zu Konflikten zwischen Änderungen kommen kann.
3. Speichern Sie die Änderungen und schließen Sie das Projekt in Flash Catalyst. Sie müssen Flash Catalyst nicht beenden.
4. Wählen Sie in Flash Builder „Projekt“ > „Arbeit am Projekt in Flash Builder fortsetzen“. Sie können auch im Kontextmenü für das Projekt „Flash Catalyst“ >„Arbeit am Projekt in Flash Builder fortsetzen“ auswählen.

In diesem Fall werden Sie zum Speichern der in Flash Catalyst vorgenommenen Änderungen aufgefordert. Durch das Speichern der Änderungen wird das neu gespeicherte Projekt wieder in Flash Builder geöffnet. Beim Import der neuen Projektversion aus Flash Catalyst wird gleichzeitig die ursprüngliche, aus Flash Builder exportierte Projektversion gelöscht.

Die Designänderungen werden dem Projekt hinzugefügt und dieses wird in Flash Builder zur Bearbeitung geöffnet.

Kapitel 11: Anpassen von Flash Builder

Adobe® Flash® Builder™ ist ein Plug-In zur Eclipse-Entwicklungsplattform. Neben den allgemeinen Voreinstellungen, die Sie für Eclipse angeben, können Sie auch Voreinstellungen für Flash Builder festlegen. Für manche Funktionen geben Sie sowohl in Eclipse als auch in den Flash Builder-Knoten des Dialogfelds „Voreinstellungen“ Benutzervorgaben an. Beispielsweise geben Sie beim Festlegen der Voreinstellungen für den Flash Builder-Debugger benutzerdefiniertes Verhalten sowohl unter dem Eclipse-Knoten „Ausführen/Debug“ als auch im Knoten „Flash Builder“ > „Debuggen“ an.

Wie Sie das Dialogfeld „Voreinstellungen“ öffnen, variiert je nach Plattform und hängt davon ab, ob Sie die eigenständige oder die Plug-In-Version von Flash Builder verwenden.

Flash Builder-Voreinstellungen angeben

1 Öffnen Sie das Dialogfeld „Voreinstellungen“.

- (Windows) Wählen Sie „Fenster“ > „Voreinstellungen“.
- (Macintosh, eigenständig) Wählen Sie „Fenster“ > „Voreinstellungen“.
- (Macintosh, Plug-In) Wählen Sie „Eclipse“ > „Voreinstellungen“.

2 Erweitern Sie den Knoten „Flash Builder“, um die Benutzervoreinstellungen anzuzeigen und anzugeben.

Adobe-Voreinstellungen

Legen Sie Voreinstellungen für Plug-In-Module von Adobe fest.

RDS-Konfiguration

Die Konfigurationsinformationen für Remote Development Server (RDS) gelten für Benutzer von ADEP (Adobe Digital Enterprise Platform) Data Services oder Adobe BlazeDS. RDS ermöglicht den Zugriff auf Data Services und BlazeDS-Ziele.

Die RDS-Standardkonfiguration ist ein Ausgangspunkt für die Verbindung mit Datendiensten. Ändern Sie die Standardkonfiguration, um den Zugriff auf Ihr Serverziel oder Ihre Datenbank zu ermöglichen.

Informationen zum Konfigurieren von RDS finden Sie in der Dokumentation zu ADEP Data Services.

Wichtig: *Ihre RDS-Konfiguration hat Auswirkungen auf die Sicherheit. Informationen zu Sicherheitsauswirkungen auf Anwendungsserver beim Angeben einer RDS-Konfiguration finden Sie in der Dokumentation zu ADEP Data Services.*

Workbench anpassen

Sie können die Workbench entsprechend Ihren individuellen Entwicklungsanforderungen anpassen. Beispielsweise können Sie anpassen, wie Elemente in der Hauptsymbolleiste erscheinen sollen, Tastaturbefehle erstellen oder Schriftarten und Farben der Benutzeroberfläche ändern.

Reihenfolge der einzelnen Symbolleisten in der Hauptsymbolleiste ändern

Flash Builder bietet Ihnen die Möglichkeit, die Reihenfolge der einzelnen Symbolleisten in der Hauptsymbolleiste zu ändern. Die einzelnen Symbolleisten sind in der Hauptsymbolleiste durch eine Doppellinie voneinander getrennt.

1. Vergewissern Sie sich, dass die Symbolleiste nicht gesperrt ist. Heben Sie im Kontextmenü der Symbolleiste „Symbolleisten sperren“ auf.
2. Setzen Sie den Mauszeiger auf die vertikale Doppellinie links neben der Symbolleiste, die Sie an eine andere Position verschieben möchten.
3. Klicken Sie auf die Doppellinie und ziehen Sie den Abschnitt nach links, rechts, oben oder unten. Lassen Sie die Maustaste los, um den Abschnitt an der neuen Position abzulegen.

Sperren Sie dann mithilfe des Kontextmenüs die Symbolleiste wieder, um versehentliche Änderungen zu vermeiden.

Tastenkombinationen ändern

1. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Tasten“ aus.
2. Wählen Sie aus der Liste im Dialogfeld „Tasten“ den Befehl aus, den Sie ändern möchten.
3. Geben Sie im Feld „Bindung“ den neuen Tastaturbefehl ein, den Sie mit dem Befehl verbinden möchten.
4. Wählen Sie im Popupmenü „Aktiv“ aus, wann der Tastaturbefehl aktiv sein soll.
5. Klicken Sie auf „Anwenden“ oder „OK“.

Eine Liste aller Tastaturbefehle finden Sie unter „Hilfe“ > „Hilfe für Tastenbelegung“.

Schriftarten und Farben ändern

Die Workbench verwendet standardmäßig Schriftarten und Farben, die vom Betriebssystem des Computers zur Verfügung gestellt werden. Sie können aber Schriftarten und Farben auf verschiedene Weisen anpassen.

1. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ > „Darstellung“ > „Farben und Schriftarten“.
2. Erweitern Sie die Kategorien „Basis“, „CVS“, „Debug“, „Textvergleich“ oder „Ordner für Ansichten und Editoren“, um die Schriftart und Farbe auszuwählen, die geändert werden sollen.

   Hinweis: *Sie können auch auf „Systemschrift verwenden“ statt auf „Ändern in“ klicken, um die Schriftart auf einen geeigneten, vom Betriebssystem gewählten Wert zu setzen. Wenn Sie diese Option beispielsweise in Windows aktivieren, verwendet Flash Builder die Schriftart, die in der Windows-Systemsteuerung in den Anzeigeeigenschaften ausgewählt wurde.*
3. Wählen Sie die gewünschte Schriftart und Farbe aus.

Einzel- und Doppelklickverhalten bestimmen

Sie können steuern, wie die Workbench auf Einzel- und Doppelklicks reagieren soll.

1. Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie „Allgemein“ aus.
2. Nehmen Sie im Abschnitt „Öffnungsmodus“ die gewünschten Einstellungen vor und klicken Sie auf „OK“.

Workbench-Voreinstellungen festlegen

Sie können Voreinstellungen für viele Aspekte der Workbench festlegen. Beispielsweise können Sie festlegen, ob Flash Builder Sie beim Starten des Programms auffordern soll, den zu verwendenden Arbeitsbereich auszuwählen. Sie können auch auswählen, welcher Editor zum Öffnen bestimmter Ressourcentypen verwendet werden soll, sowie verschiedene Optionen zum Ausführen und Debuggen von Anwendungen festlegen.

Die Flash Builder-Voreinstellungen beziehen sich nur auf den aktuellen Arbeitsbereich. Sie können die Workbench-Voreinstellungen aber auch exportieren und dann in einen anderen Arbeitsbereich importieren. Dies ist praktisch, wenn Sie mehrere Arbeitsbereiche verwenden oder wenn Sie die Workbench-Voreinstellungen an Kollegen weitergeben möchten.

Sie können auch Voreinstellungen für einzelne Projekte in einem Arbeitsbereich festlegen. Beispielsweise könnten Sie separate Compiler- oder Debugoptionen für die einzelnen Flex-Projekte festlegen.

Voreinstellungen für die Flash Builder-Workbench festlegen

1. Öffnen Sie das Fenster „Voreinstellungen“.
2. Erweitern Sie den Eintrag „Allgemein“ und wählen Sie die gewünschte Kategorie der Workbench-Voreinstellungen aus und ändern Sie sie entsprechend Ihrem Bedarf ab.
3. Klicken Sie auf „OK“.

Flash Builder-Voreinstellungen

Flash Builder

Warnung anzeigen, bevor alte Flash Builder-Projekte aktualisiert werden

Flash Builder aktualisiert Projekte, die mit einer früheren Version von Flash Builder erstellt wurden. Flash Builder aktualisiert die Projektdateien und die Struktur so, dass sie mit der aktuellen Struktur von Flash Builder-Projekten übereinstimmt. Der Zugriff auf ein konvertiertes Projekt ist mit der früheren Version von Flash Builder nicht mehr möglich.

Standardmäßig wird in Flash Builder vor der Konvertierung eine Warnmeldung angezeigt. Deaktivieren Sie diese Option, wenn Konvertierungen automatisch ausgeführt werden sollen.

Daten/Dienste

Die folgenden Benutzervoreinstellungen sind für Daten/Dienste verfügbar. Diese Voreinstellungen gelten für alle Projekte in Ihrer Flash Builder-Entwicklungsumgebung.

Sie können diese Voreinstellungen im Projekt überschreiben. Wählen Sie „Projekt“ > „Eigenschaften“ > „Daten/Dienste“, um die hier angegebenen Voreinstellungen zu überschreiben.

Codegenerierung

Mit dem Standard-Codegenerierungsdienstprogramm von Flash Builder generieren Sie den Zugriff auf Datendienste. Mithilfe von Flash Builder-Erweiterbarkeitsfunktionen können Sie eigene Dienstprogramme zur Codegenerierung erstellen und sie als Plug-Ins zu Flash Builder hinzufügen. Weitere Informationen finden Sie in der Flash Builder Extensibility API-Referenz.

Erweiterungen zur Codegenerierung, die Sie als Plug-In zu Flash Builder hinzufügen, können über das Kombinationsfeld „Codegenerator“ aufgerufen werden.

Service Manager für die Verwendung einer einzelnen Dienstinstanz (Singleton) während der Codegenerierung aktivieren
Diese Option ist standardmäßig nicht aktiviert. Während der Codegenerierung erstellt jede Clientanwendung in einem Projekt ihre eigene Datendienstinstanz.

Aktivieren Sie diese Option, wenn Sie eine einzige Dienstinstanz wünschen, die von allen Clientanwendungen im Projekt gemeinsam verwendet wird.

Diese Option ist nur verfügbar, wenn Sie das Standard-Flash Builder-Dienstprogramm zur Codegenerierung angeben.

Debuggen

Die folgenden Optionen zum Debuggen sind in Flash Builder automatisch aktiviert. Weitere Optionen, die sich auf eine Debugsitzung auswirken, finden Sie, wenn Sie folgende Optionen wählen:

- „Voreinstellungen“ > „Allgemein“ > „Webbrowser“
- „Voreinstellungen“ > „Ausführen/Debug“

Warnen, wenn versucht wird, mehrere Debugsitzungen zu starten

Bei manchen Plattform/Browser-Kombinationen sind keine gleichzeitigen Debugsitzungen möglich. Wenn Sie eine zweite Debugsitzung starten, wird die ursprüngliche Debugsitzung beendet, oder die Verbindung wird unterbrochen.

Lassen Sie diese Option aktiviert, wenn Flash Builder eine Warnmeldung anzeigen soll, wenn Sie eine zweite Debugsitzung starten.

Warnen, dass nach dem Start ein SecurityError auftreten kann

In manchen Fällen zeigt ein Webbrowser eine Sicherheitsfehlermeldung an, weil er nicht in der Lage ist, die Sicherheitsdatei von Flash Player zu lesen. In den meisten Fällen wird das Sicherheitsproblem durch Neustart des Webbrowsers behoben.

Lassen Sie diese Option aktiviert, wenn Flash Builder eine Warnmeldung in Bezug auf Sicherheitsfehler dieser Art anzeigen soll.

Siehe Technische Hinweise zur Flash Player-Sicherheitswarnung.

Getter-Funktionen automatisch aufrufen

Beim Durchlaufen einer Debugsitzung werden Variablen, die Accessor-Funktionen (Getter) darstellen, automatisch überprüft. Dieses Verhalten ist im Allgemeinen während einer Debugsitzung nützlich.

Deaktivieren Sie diese Option, wenn beim Durchlaufen einer Debugsitzung Variablen, die Accessor-Funktionen darstellen, nicht automatisch überprüft werden sollen.

Versionsverwaltung für die eigenständige Flash Player-Version

Wenn Sie in Flash Builder eine Anwendung starten oder debuggen, können Sie angeben, dass der ausführbare Flash Player verwendet werden soll.

Wählen Sie hierzu im Dialogfeld „Voreinstellungen“ die Optionen „Flash Builder“ > „Debuggen“. Definieren Sie anschließend den Speicherort des ausführbaren Flash Player, mit dem Flash Builder die Anwendung debuggen oder starten soll.

Gerätekonfigurationen

Für die Vorschau von Gerätedisplaygrößen in der Designansicht und beim Starten von Anwendungen auf dem Desktop mit dem AIR Debug Launcher verwendet Flash Builder Gerätekonfigurationen. Siehe „Startkonfigurationen verwalten“ auf Seite 123.

Flash Builder ist mit Standardkonfigurationen für die Nexus One, Droid, Apple iPad, Apple iPhone, HTC und Samsung Android ausgestattet. Die Standardkonfigurationen können nicht bearbeitet werden.

Editoren in Flash Builder

Allgemein

Flash Builder enthält Benutzeroptionen für Codefaltung, wodurch Probleme beim Programmieren protokolliert werden, und für die Positionierung des Carets unter Wörtern.

Geschweifte Klammern

Flash Builder bietet Benutzeroptionen zum Einrücken, Einfügen und Hervorheben geschweifter Klammern, die Codeblöcke darstellen.

Codehilfe

Bei der Verwendung des MXML- oder ActionScript-Quelleditors zeigt Flash Builder Codehinweise an, um Ihnen beim Vervollständigen von Codeausdrücken zu helfen. Die Codehilfe umfasst eine Hilfe zum Auswählen von empfohlenen Klassen, Eigenschaften und verfügbaren Ereignissen.

- **Auto-Aktivierung aktivieren**

  Deaktivieren Sie die Option „Auto-Aktivierung aktivieren“, wenn bei der Eingabe keine Codehinweise automatisch angezeigt werden sollen. Wenn Sie diese Option deaktivieren, können Sie Codehinweise abrufen, indem Sie Strg/Befehl+Leertaste drücken.

- **Danach aktivieren**

  Geben Sie an, nach wie vielen Millisekunden ab Beginn der Eingabe Codehinweise angezeigt werden sollen. Der Standardwert ist 100 Millisekunden.

- **Verwenden weiterer benutzerdefinierter Auslöser**

  Codehinweise können nicht nur mit Strg+Leertaste aufgerufen werden, sondern auch über benutzerdefinierte Tastenfolgen (sog. Auslöser).

  Drücken Sie die Tasten, die in MXML- oder ActionScript-Code als Auslöser für die Codehinweise dienen sollen. Es braucht nur eine Taste des Auslösers gedrückt zu werden, um einen Codehinweis aufzurufen. Angenommen, Sie legen die Tasten `ab` als Auslöser fest. Nun brauchen Sie nur `a` oder `b` zu drücken, um die Codehilfe aufzurufen.

Verwandte Themen

„Tools zur Codeentwicklung in Flash Builder“ auf Seite 49

Einrückung

Mit Flash Builder können Einrückungen auch durch Tabstopps statt durch Leerzeichen erzeugt werden. Außerdem können neue Zeilen bei der Eingabe automatisch eingerückt werden.

Allgemeine Voreinstellungen des Flash Builder-Editors

Standardmäßig bietet Flash Builder Codeeinklappung und automatisches Einrücken von Zeilen beim Eingeben oder Einfügen von Text in den Quelleditor. Sie können die Standardeinstellungen für diese Funktionen deaktivieren.

Allgemeine Voreinstellungen des Eclipse-Editors

Sie können auf zusätzliche allgemeine Voreinstellungen für Editoren zugreifen, indem Sie „Voreinstellungen“ > „Allgemein“ > „Editoren“ wählen

Code in ActionScript

Beim Bearbeiten von ActionScript-Dateien im Quelleditor bietet Flash Builder Standardfunktionen zum Umbrechen und Strukturieren von Code. Standardmäßig platziert Flash Builder alle wichtigen Importanweisungen in den Kopf einer ActionScript-Datei und entfernt Importanweisungen, für die im Code keine Verweise vorhanden sind.

Standardmäßig wird die ActionScript-Referenzdokumentation, wenn verfügbar, mit Codehinweisen oder durch Bewegen des Mauszeigers über ein Sprachelement angezeigt.

Jede dieser Standardfunktionen kann deaktiviert werden.

Beim Eingeben von ActionScript-Code in den Flash Builder-Editoren zeigt die Inhaltshilfe eine Liste von Optionen zur Vervollständigung der Codeausdrücke an. Sie können den teilweise eingegebenen ActionScript-Code über entsprechende Auslösertasten automatisch vervollständigen lassen. Zu diesem Zweck ist Flash Builder bereits mit Auslösertasten vorkonfiguriert. Diese können Sie jedoch ändern. Sie können den Code auch nach Drücken der Leertaste einfügen.

Inhaltshilfe durchlaufen

Beim Eingeben von ActionScript-Code drücken Sie mehrmals Strg/Befehl+Leertaste, um zwischen den Filtern für die angezeigten Codehinweise zu wechseln.

Sie können festlegen, welche Codehinweise angezeigt sowie in welcher Reihenfolge sie durchlaufen werden sollen. Um die Standardeinstellung zu ändern, gehen Sie wie folgt vor:

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie die Optionen „Flash Builder“ > „Editoren“ > „ActionScript-Code“ > „Inhaltshilfe - Zyklierung“.

2 Wählen Sie den gewünschten Hinweistyp und klicken Sie auf „Nach oben“ oder „Nach unten“, um die Reihenfolge der Zyklierung zu ändern.

   Der erste Hinweistyp in der Liste ist der zuerst angezeigte Codehinweis.

Einrückung

Flash Builder rückt standardmäßig Paketinhalt, Funktionsdeklarationen und Switch-Anweisungen ein. Jede dieser Standardfunktionen kann deaktiviert werden.

Verwandte Themen

„Tools zur Codeentwicklung in Flash Builder“ auf Seite 49

„Inhaltshilfe“ auf Seite 49

Codevorlagen

Sie können den vordefinierten Stubcode, den Flash Builder beim Erstellen von Methoden oder Get- und Set-Accessor-Funktionen, Generieren von Ereignisprozeduren oder Überschreiben von Methoden erzeugt, individuell anpassen.

Flash Builder hält vordefinierte Codevorlagen bereit, mit deren Hilfe Sie schnell MXML-, ActionScript- und CSS-Code erstellen können. Sie können Codevorlagen erstellen und bearbeiten. Sie können die Codevorlagen auch importieren und exportieren.

Weitere Informationen zum Einsatz, Erstellen und Bearbeiten von Codevorlagen finden Sie unter „Codevorlagen erstellen und bearbeiten“ auf Seite 58.

Designmodus

Im Designmodus des Quelleditors von Flash Builder stehen die folgenden Benutzervoreinstellungen zur Verfügung:

- Mit dem Design verknüpfte Ansichten automatisch anzeigen

  Wenn Sie in Flash Builder in den Designmodus des Quelleditors wechseln, werden mit dem Design verknüpfte Ansichten automatisch geöffnet. Beispiele:. „Eigenschaften“, „Status“, „Komponenten“ und „Aussehen“.

  Deaktivieren Sie dieses Kontrollkästchen, wenn Sie bestimmen möchten, welche Ansichten im Designmodus geöffnet werden sollen.

- Objektausrichtung aktivieren

  Im Designmodus von Flash Builder werden hinzugefügte oder neu angeordnete Komponenten mithilfe eines unsichtbaren Rasters automatisch ausgerichtet.

  Deaktivieren Sie dieses Kontrollkästchen, wenn Sie die Anordnung der Komponenten lieber ohne Raster vornehmen möchten.

- Skins beim Öffnen von Dateien rendern

  Im Designmodus werden im Designbereich befindliche Komponenten mit einer Skin gezeichnet.

  Deaktivieren Sie dieses Kontrollkästchen, wenn die Skins von Komponenten nicht im Designmodus gerendert werden sollen.

- Datenbindungsausdrücke ausblenden

  Im Designmodus geben Datenbindungsausdrücke im Allgemeinen die Form oder Größe der Daten, an die die Ausdrücke gebunden sind, nicht präzise dar.

  Aktivieren Sie dieses Kontrollkästchen, wenn im Designmodus des Editors Komponenten, die an Datenbindungsausdrücke gebunden sind, präziser wiedergegeben werden sollen.

- Verweise beim Ändern von IDs in der Flex-Eigenschaftenansicht aktualisieren

  Wenn Sie in der Eigenschaftenansicht eine Komponenten-ID ändern, kann sich dies auf Code in Ihrem Arbeitsbereich auswirken, der auf diese ID verweist.

  Aktivieren Sie dieses Kontrollkästchen, wenn Flash Builder automatisch alle Verweise auf Komponenten-IDs aktualisieren soll, wenn Sie in der Eigenschaftenansicht eine Komponenten-ID bearbeiten.

- MXML-Dateien immer im Codeeditor öffnen

  Wenn Sie eine MXML-Datei im Paketexplorer öffnen, wird sie immer im Quellmodus des Editors geöffnet.

  Wenn Sie diese Voreinstellung deaktivieren, wird die MXML-Datei im zuletzt vom Editor verwendeten Modus geöffnet. Beispiel: Die MXML-Datei wurde zuletzt im Designmodus geöffnet und anschließend geschlossen. Beim nächsten Öffnen der MXML-Datei wird sie wieder im Designmodus geöffnet.

MXML-Code

Beim Bearbeiten von MXML-Dateien im Quelleditor bietet Flash Builder Standardfunktionen zum Vervollständigen von Code. Jede dieser Funktionen kann deaktiviert werden.

Inhaltshilfe durchlaufen

Flash Builder bietet außerdem das Standardverhalten der Inhaltshilfe zum Bearbeiten von MXML-Dateien an. Wenn die Inhaltshilfe verfügbar ist, zeigt Flash Builder Hinweise zu verfügbaren Sprachelementen an, die in den Code eingefügt werden können. Drücken Sie mehrmals Strg/Befehl+Leertaste, um durch die in der Inhaltshilfe angezeigten Elemente zu wechseln.

Sie können festlegen, welche Codehinweise angezeigt sowie in welcher Reihenfolge sie durchlaufen werden sollen. Um die Standardeinstellung zu ändern, gehen Sie wie folgt vor:

1 Öffnen Sie das Dialogfeld „Voreinstellungen“ und wählen Sie die Optionen „Flash Builder“ > „Editoren“ > „MXML-Code“ > „Inhaltshilfe - Zyklierung“.

2 Geben Sie an, welche Hinweistypen angezeigt werden sollen.

3 Wählen Sie den gewünschten Hinweistyp und klicken Sie auf „Nach oben“ oder „Nach unten“, um die Reihenfolge der Zyklierung zu ändern.

   Der erste Hinweistyp in der Liste ist der zuerst angezeigte Codehinweis.

Verwandte Themen

„Inhaltshilfe“ auf Seite 49

„Tools zur Codeentwicklung in Flash Builder“ auf Seite 49

Formatierung

Flash Builder formatiert standardmäßig automatisch generierten MXML-Code. So formatiert Flash Builder MXML-Code, der automatisch generiert wird, wenn Sie in der Designansicht eine Komponente hinzufügen oder mithilfe von Flash Builder-Werkzeugen eine Verbindung zu einem Datendienst herstellen und auf Daten zugreifen.

Sie können die Reihenfolge und Gruppierung der Attribute im Dialogfeld für die Formatierungsvoreinstellungen ändern.

Für manuell erstellten Code steht die Codeformatierung nicht zur Verfügung. Um einen manuell erstellten Codeabschnitt zu formatieren, wählen Sie „Quelle“ > „Formatierung anwenden“ (Strg+Umschalt+F). Der Code wird dann formatiert, wobei nicht berücksichtigt wird, ob die Option „Anordnung der MXML-Attribute beibehalten“ aktiviert wurde oder nicht.

Einrückung

Standardmäßig verwendet Flash Builder Tabulatoren zum Einrücken von Code. Sie können die Standardeinstellung ändern und Leerzeichen verwenden. Der Standardwert zum Einrücken mit Tabulatoren bzw. Leerzeichen ist 4

Wählen Sie im Dialogfeld „Voreinstellungen“ die Einträge „ActionScript“ und „MXML“, um eine Vorschau der Einrückungseinstellungen anzuzeigen. Sie können die Einrückungsheuristik auch anpassen.

Syntaxhervorhebung

Flash Builder bietet standardmäßige farbige Syntaxhervorhebung und Texthervorhebung für ActionScript-, CSS- und MXML-Dateien. Sie können diese Standardfunktionen überschreiben. Erweitern Sie den Knoten der gewünschten Sprache und wählen Sie eine Sprachfunktion aus, deren Standardeinstellung Sie überschreiben möchten.

Allgemeine Voreinstellungen des Eclipse-Editors sind ebenfalls verfügbar. Siehe:

- „Voreinstellungen“ > „Allgemein“ > „Texteditoren“
- „Voreinstellungen“ > „Allgemein“ > „Farben und Schriftarten“

Dateizuordnungen

Sie können festlegen, dass MXML- und ActionScript-Dateien standardmäßig mit Flash Builder geöffnet werden sollen. Die Angabe einer Dateizuordnung überschreibt die Einstellungen des Betriebssystems.

1 Öffnen Sie das Dialogfenster „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Dateizuordnungen“.

2 Wählen Sie „ActionScript-Dateien (*.as)“ und „MXML-Dateien (*.mxml)“, um die Dateizuordnungen festzulegen.

Dateiausschlüsse

Sie können festlegen, welche Dateitypen bei der Projektkompilierung nicht von Flash Builder in den Ausgabeordner kopiert werden sollen.

1 Öffnen Sie das Dialogfenster „Voreinstellungen“ und wählen Sie „Flash Builder“ > „Dateiausschlüsse“.

2 Sie können neue Dateinamenerweiterungen und Dateinamen hinzufügen oder vorhandene aus der Liste entfernen.

3 Zum Hinzufügen einer neuen Dateinamenerweiterung oder eines Dateinamens klicken Sie auf „Neu“ und geben Sie die gewünschte Erweiterung oder Datei an.

4 Klicken Sie zum Wiederherstellen der Vorgabenliste von Erweiterungen auf „Standard wiederherstellen“.

Dateivorlagen

Für das Erstellen von neuen MXML-, ActionScript- und CSS-Dateien verwendet Flash Builder Dateivorlagen. Sie können die von Flash Builder zum Erstellen dieser Dateien verwendeten Vorlagen anpassen. Ferner können Sie Vorlagendateien importieren und exportieren.

Wählen Sie im Bereich „Dateitypen“ eine Vorlage für einen Dateityp aus. Klicken Sie auf „Bearbeiten“, um die Vorlage zu modifizieren. Deaktivieren Sie „Neue Dateien automatisch einrücken“, wenn Sie beim Erstellen von Dateien die Einrückungsvoreinstellungen von Flash Builder nicht übernehmen möchten.

Sie können die Flash Builder-Voreinstellungen für die Einrückung ändern. Wechseln Sie zu „Voreinstellungen“ > „Flash Builder“ > „Einrückung“.

Verwandte Themen

„Dateivorlagen anpassen“ auf Seite 63

„Einrückung“ auf Seite 270

FlexUnit

Standardstartmodus

Beim Ausführen eines FlexUnit-Tests wird Flash Builder standardmäßig im Modus „Debuggen“ gestartet.

Mit dieser Voreinstellung bestimmen Sie den Startmodus von Flash Builder bei FlexUnit-Tests.

Wählen Sie aus den folgenden Standardstartkonfigurationen für Flash Builder:

- Modus „Ausführen“

  Standardmäßig wird die Flash-Perspektive gestartet.

- Modus „Debuggen“

  Standardmäßig wird die Flash-Debug-Perspektive gestartet.

- Modus „Profilieren“

  Standardmäßig wird die Perspektive „Flash-Profil“ gestartet.

Mit den Eclipse-Voreinstellungen für den Startvorgang und für Perspektiven konfigurieren Sie Startmodi. Siehe:

- „Voreinstellungen“ > „Ausführen/Debug“ > „Startvorgang“
- „Voreinstellungen“ > „Ausführen/Debug“ > „Perspektiven“

Benutzerdefinierter Name der Anwendung
Geben Sie den vollständig qualifizierten Namen der benutzerdefinierten FlexUnit-Anwendungsdatei an. Wenn die benutzerdefinierte FlexUnit-Anwendungsdatei sich nicht im ausgewählten Flex-Projekt befindet, generiert Flash Builder automatisch eine standardmäßige FlexUnit-Anwendungsdatei.

Portnummer
Gibt die Portnummer an, die verwendet werden soll, um eine Verbindung von Flash Builder mit der FlexUnit-Anwendung auf dem Mobilgerät herzustellen. Die Portnummer ist ein hartcodierter Wert 8765.

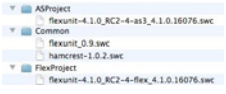
Alternatives Framework verwenden
Standardmäßig werden die SWC-Dateien des Frameworks, die in Flash Builder verfügbar sind, dem Erstellungspfad von FlexUnit hinzugefügt.

Wenn Sie ein anderes FlexUnit-Framework angeben möchten, wählen Sie „Anderes Framework verwenden“. Navigieren Sie zum Speicherort mit dem Ordner des Frameworks, das Sie verwenden möchten.

Wenn Sie eine FlexUnit-TestCase-Klasse oder -Suite für das ausgewählte Projekt erstellen, fügt Flash Builder automatisch die im Ordner des alternativen Frameworks vorhandenen SWC-Dateien dem FlexUnit-Erstellungspfad hinzu.

Wenn Sie einen Ordner für ein alternatives FlexUnit-Framework erstellen, empfiehlt sich für die SWC-Dateien folgende Ordnerstruktur:

- `flexunit-x.x-as3_x.x.swc`-Dateien im Ordner „ASProject“. Bei ActionScript-Projekten werden die SWC-Dateien in diesem Ordner automatisch in den jeweiligen Erstellungspfad kopiert.
- `flexunit-x.x-flex_x.x.swc`-Dateien im Ordner „FlexProject“. Bei Flex-Projekten werden die SWC-Dateien in diesem Ordner automatisch in den jeweiligen Erstellungspfad kopiert.
- Alle anderen SWC-Dateien, die für das Projekt erforderlich sind oder von denen der FlexUnit-Code abhängig ist (die sich noch nicht bereits im Erstellungspfad befinden), kommen in den Ordner „Common“. Bei ActionScript- und Flex-Projekten werden die SWC-Dateien in diesem Ordner automatisch in den jeweiligen Erstellungspfad kopiert.



Wenn Flash Builder die empfohlene Ordnerstruktur nicht finden kann, wird der ausgewählte Alternativordner unabhängig davon, ob es sich um ein Flex- oder ein ActionScript-Projekt handelt, angelegt.

Adobe Community-Experte Paul Robertson schrieb einen Artikel über das Erstellen eines benutzerdefinierten FlexUnit-Framework.

Verwandte Themen

„FlexUnit-Testumgebung“ auf Seite 178

„Flash Builder-Perspektiven und -Ansichten“ auf Seite 7

Installierte Flex SDKs

Flash Builder 4.6 installiert das Flex 4.6 SDK und das Flex 3.6 SDK. Für Projekte, in denen kein SDK angegeben ist, verwendet Flash Builder das Flex 4.6 SDK.

Sie können zusätzliche SDKs hinzufügen, SDKs entfernen und angeben, welches SDK bei Projekten, die kein SDK angeben, standardmäßig verwendet werden soll. Flash Builder fordert Sie beim Importieren von Projekten, die in einer früheren Version von Flash Builder erstellt wurden, auf, das zu verwendende SDK anzugeben. Sie können diese Aufforderung für das Importieren von Projekten deaktivieren.

Die folgenden SDK-Typen stehen zur Verfügung:

Kostenloses Adobe Flex SDK Veröffentlichte Versionen von offiziellen Adobe-Produkten. Diese SDKs enthalten eine Mischung von Open-Source- und Closed-Source-Komponenten.

Open-Source-Flex SDK Für Benutzer, die ein Paket nur mit Open-Source-Code wünschen.

Adobe Add-ons zum Open-Source-Flex SDK Dieses Paket enthält den gesamten Inhalt des Adobe Flex SDK, der nicht im Open-Source-Flex SDK enthalten ist. Wenn Sie diese Datei herunterladen, bringen Sie das Open-Source-Flex SDK auf den gleichen Stand wie das Adobe Flex SDK.

Flex SDK hinzufügen

1 Laden Sie ein Flex SDK auf einen temporären Speicherort Ihres Computers herunter.

Lesen Sie auf den Downloadseiten die Erläuterungen zu den unterschiedlichen Buildtypen, Anforderungen und Lizenzinformationen.

- Flex 4.6 SDKs herunterladen
- Flex 4.5 SDKs herunterladen.
- Flex 4 SDKs herunterladen.
- Flex 3 SDKs herunterladen.

Hinweis: *Wenn Sie eine stabile, getestete Version wünschen, laden Sie einen Milestone-Realeasebuild herunter. Ein Milestone-Releasebuild wurde vom Entwicklungsteam zur Hauptversion erklärt. Darin ist eine signierte Version der Flex-Framework-RSLs enthalten (RSL-Caching erforderlich).*

2 Die herunterzuladenden Flex SDKs sind ZIP-Dateien. Entpacken Sie die ZIP-Datei in einem neuen Ordner auf Ihrem Rechner.

Es wird empfohlen, den Ordner der obersten Ebene mit der SDK-Versionsnummer zu benennen.

Die übliche Vorgehensweise ist, die SKDs im Flash Builder-Installationsordner des folgenden Pfads zu platzieren:

`<Flash Builder-Installationsordner>/`sdks/

3 Wählen Sie in den Flash Builder-Voreinstellungen „Flash Builder“ > „Installierte Flex SDKs“.

4 Klicken Sie auf „Hinzufügen“ und wechseln Sie zum entpackten Flex SDK. Klicken Sie auf „Anwenden“.

5 Öffnen Sie für jedes Projekt, das das Flex SDK verwendet, das Dialogfeld „Voreinstellungen“ und machen Sie es zum Standard-SDK für dieses Projekt. Klicken Sie auf „OK“.

Netzwerküberwachung

Auf der Voreinstellungsseite „Netzwerküberwachung“ werden die Ports aufgeführt, an denen die Netzwerküberwachung Ereignisse erfasst und HTTP-Anforderungen abhört.

Standardmäßig löscht die Netzwerküberwachung beim Start alle Überwachungseinträge. Diese Voreinstellung kann deaktiviert werden.

Sie können auch die folgenden Voreinstellungen aktivieren:

- Überwachung beim Start unterbrechen
- SSL-Sicherheitsprüfungen ignorieren

  Das Aktivieren dieser Option ist nützlich, wenn Sie Netzwerkverkehr von einem selbst signierten Server überwachen.

Profiler

Standardmäßig sind sowohl „Speicher-Profiling“ als auch „Leistungs-Profiling“ aktiviert.

- Speicher-Profiling

  Speicher-Profiling wird normalerweise verwendet, um zu prüfen, wie viel Speicher jedes Objekt bzw. jeder Objekttyp in der Anwendung belegt. Durch Verwenden der Speicher-Profilingdaten können Sie die Größe der Objekte reduzieren, die Anzahl der erstellten Objekte verringern oder Objekte durch Entfernen der Verweise auf sie mithilfe des Garbagekollektors bereinigen lassen.

  Speicher-Profiling beansprucht viel mehr Speicher als Leistungs-Profiling und kann die Leistung Ihrer Anwendung beeinträchtigen.

- Leistungs-Profiling

  Sie verwenden Leistungs-Profiling normalerweise, um in der Anwendung nach Methoden zu suchen, die nur langsam ausgeführt werden und die optimierungswürdig sind.

Sie können auch die folgenden Voreinstellungen für den Profiler angeben:

- Verbindungen

  Hier können Sie die Nummer des Ports angeben, auf dem Flash Builder die profilierte Anwendung erwartet. Die Standardportnummer ist 9999. Sie können hierfür nicht Port 7935 wählen, da dieser vom Debugger verwendet wird.

- Ausschlussfilter

  Hiermit definieren Sie die Standardpakete, die von den Profileransichten ausgeschlossen sind.

- Einschlussfilter

  Hiermit definieren Sie die Standardpakete, die in den Profileransichten eingeschlossen werden sollen. Alle anderen Pakete werden ausgeschlossen.

- Objektverweise

  Gibt die Anzahl der Rückverweispfade zu den Instanzen eines anzuzeigenden Objekts an. Mithilfe der Rückverweispfade können Sie ermitteln, ob ein Pfad einen Rückverweis zum Garbagekollektor (GC-Stamm) hat. Eine Instanz, deren Freigabe erwartet wurde, die aber Verweise auf den GC-Stamm aufweist, zeigt ein Speicherleck an.

  Standardmäßig zeigt der Profiler zehn Rückverweispfade an. Sie können eine andere Maximalzahl anzuzeigender Rückverweispfade angeben oder angeben, dass alle Rückverweispfade angezeigt werden sollen.

- Player/Browser

Sie können angeben, welcher eigenständige Adobe Flash Player und welcher Webbrowser zum Profilieren externer Anwendungen verwendet werden soll. Verwenden Sie eine Debuggerversion von Flash Player, um die Anwendung erfolgreich zu profilieren.

Beim Profilieren externer Anwendungen verwendet Flash Builder standardmäßig die folgenden Flash Player:

- URL zu einer SWF-Datei

  Flash Builder startet die SWF-Datei der Anwendung mithilfe des Standardbrowsers für das System.

- Speicherort der SWF-Datei im Dateisystem

  Flash Builder öffnet die Anwendung mit der Standard-Debuggerversion des eigenständigen Flash Players.

Verwandte Themen

„Der Profiler“ auf Seite 166

„Profiler-Ansichten“ auf Seite 149

Zielplattformen

Für die Entwicklung von Mobilanwendungen für die Android-Plattform installiert Flash Builder erforderliche Teile des Android SDK.

Primäre Netzwerkschnittstelle auswählen

Beim Debuggen der Anwendung über das Netzwerk kann der Hostcomputer mit mehreren Netzwerkschnittstellen gleichzeitig verbunden sein. Sie können jedoch eine primäre Netzwerkschnittstelle für das Debuggen im Android APK-Paket oder im Debug-iOS-Paket auswählen.

Verwandte Themen

Ausführen und Debuggen von Mobilanwendungen

Flash Builder erweitern

Die Flash Builder Extensibility-API ermöglicht es Ihnen, Flash Builder so zu erweitern, dass benutzerdefinierte Komponenten unterstützt werden. Mithilfe der Flash Builder Extensibility-API können Sie die Designansicht und die Eigenschaftenansicht in Flash Builder erweitern, sodass sie benutzerdefinierte Komponenten ordnungsgemäß verarbeiten.

Antworten zu häufig gestellten Fragen zu Entwurfsansichtserweiterungen finden Sie unter Flash Builder Design View extension FAQ.

Darüber hinaus können Sie die Unterstützung von Flash Builder für die Dienste, die über den Assistenten „Dienste“ verfügbar sind, erweitern.

Weitere Informationen finden Sie in der Flash Builder Extensibility API Reference.